Engine cylinder pressure reconstruction using crank kinematics and recurrently-trained neural networks

Article (Accepted Version)

ENGINE CYLINDER PRESSURE RECONSTRUCTION USING CRANK KINEMATICS AND RECURRENTLY-TRAINED NEURAL NETWORKS

by

C. Bennett, J.F. Dunne*, S. Trimby, and D. Richardson †

Department of Engineering and Design
School of Engineering and Informatics
University of Sussex, Falmer, Brighton, BN1 9QT, UK.
* Corresponding author: E-mail: j.f.dunne@sussex.ac.uk

† Powertrain Research & Technology, Jaguar Land Rover
Viscount 2 (W11/8 Unit C2), Millburn Hill Road,
Cannon Park, Coventry, CV4 7HS

ABSTRACT

A recurrent non-linear autoregressive with exogenous input (NARX) neural network is proposed, and a suitable fully-recurrent training methodology is adapted and tuned, for reconstructing cylinder pressure in multi-cylinder IC engines using measured crank kinematics. This type of indirect sensing is important for cost effective closed-loop combustion control and for On-Board Diagnostics. The challenge addressed is to accurately predict cylinder pressure traces within the cycle under generalisation conditions: i.e. using data not previously seen by the network during training. This involves direct construction and calibration of a suitable inverse crank dynamic model, which owing to singular behaviour at top-dead-centre (TDC), has proved difficult via physical model construction, calibration, and inversion. The NARX architecture is specialised and adapted to cylinder pressure reconstruction, using a fully-recurrent training methodology which is needed because the alternatives are too slow and unreliable for practical network training on production engines. The fully-recurrent Robust Adaptive Gradient Descent (RAGD) algorithm, is tuned initially using synthesised crank kinematics, and then tested on real engine data to assess the reconstruction capability. Real data is obtained from a 1.125 litre, 3-cylinder, in-line, direct injection spark ignition (DISI) engine involving synchronised measurements of crank kinematics and cylinder pressure across a range of steady-state speed and load conditions. The paper shows that a RAGD-trained NARX network using both crank velocity and crank acceleration as input information, provides fast and robust training. By using the optimum epoch identified during RAGD training, acceptably accurate cylinder pressures, and especially accurate location-of-peak-pressure, can be reconstructed robustly under generalisation conditions, making it the most practical NARX configuration and recurrent training methodology for use on production engines.

27 main-section pages (double spaced) 31 references Figures 1 – 16 No appendices

1. INTRODUCTION

Knowledge of the cylinder pressure traces arising in an internal combustion (IC) engine can provide crucial information for feedback control of combustion to improve thermal efficiency, reduce CO_2 emissions, and to reduce harmful emissions. Multi-cylinder gasoline engines in particular have, between cycles and cylinders, significant variability in part-load volumetric efficiency and in-cylinder air motion during part-throttled operation. It is well known that the ability to control fuel injection and ignition-timing in order to balance this variability, can dramatically improve efficiency, i.e. using spark timing to phase the non-knocking combustion at light load closer to maximum efficiency [1], or to a lesser extent, by allowing closer operation to the knock limit [2]. In addition, certain features required for On Board Diagnostics could also be improved were pressure traces available for misfire detection, and for in-vehicle calibration by using the cylinder pressure trace for torque estimation [3]. Previous approaches using other on-board sensors, include the use of engine block vibration [4][5], crank kinematics [6][7], and spark ignition ionisation current [8].

The use of in-cylinder pressure transducers is common under test conditions, but still relatively rare on production engines, owing to cost and durability issues. Indirect cylinder pressure reconstruction, by processing information from existing sensors performing other functions, can provide an attractive low-cost alternative to pressure traces for combustion control on production engines. For example, crank position sensors, and knock sensors (i.e. accelerometers), are now fitted as standard on gasoline engines, and have proven durability and acceptable cost. In attempting to reconstruct cylinder pressure using indirectly-sensed information, a causal mathematical model is assumed, linking cylinder pressure to the sensor output. The challenge then, for indirect reconstruction, is either to construct the causal model and then invert it, or to construct an inverse model directly, and

then calibrate it. The hope is that when the inverse model is fed with the corresponding sensor signal, the output is precisely the required cylinder pressure.

Various indirect reconstruction methods have been attempted over the past 20 years. These fall broadly into four categories: i) via inverted physical crank dynamic models, ii) via inverted engine block vibration or acoustic transfer functions, iii) via Artificial Neural Networks (ANNs), trained as a non-parametric inverse model to handle previously unseen (generalisation) data, or iv) using spark plug ionisation current (for SI engines) which exploits the effect of combustion by building a relationship to cylinder pressure.

The first three approaches ultimately involve construction of a non-linear relationship between the measured response and the corresponding cylinder pressure. Since the converse (causal) relationship, between cylinder pressure and measured response, is generally amplitude and frequency dependent (i.e. a function of speed and load), constructing the inverse model is very problematic. This is one reason why neural networks are particularly attractive because they provide a powerful and efficient approach to nonlinear system identification. The use of spark plug ionisation current has yet to be fully verified.

One of the difficulties with indirect reconstruction is that the sensed signal may be heavily influenced by sources other than cylinder pressure. For example reconstruction via crank acceleration assumes that the gas pressure acting on the piston crown drives the slider-crank, causing acceleration of the crankshaft. Crank acceleration is however also a function of crank angle, speed, instantaneous inertia, and friction. At certain engine speeds, torsional vibration may also be significant. Similarly, the use of vibration and acoustic signals assume that combustion pressure excites engine-block or cylinder-head vibration, which can be measured directly using accelerometers positioned on the block or head (such as used for knock control). However, engine block vibration can be caused by

piston slap, intake and exhaust valve events, and fuel injector actuation. Fortunately these events occur at crank angles well separated from the combustion event for the cylinder under consideration, except in 4-stroke engines with more than four cylinders, there may be problematic overlap.

Early contributions to indirect reconstruction via crank kinematics include using an electrical analogue crank dynamics model to link crank speed fluctuation to indicated torque [9], where the method was verified on a 1.5 litre gasoline I4 engine with speeds ranging from 1500 to 3500 rpm. Ten years later, a radial basis function (RBF) network proposed in [10], was applied in [11] using data from a 2.5 litre I4 diesel engine at 39 test points, between 1000 rpm and 2600 rpm at 20 Nm, to show that P_{max} could often be reconstructed within 5%, and θ_{max} within ± 2°. Use of crank speed and position in [12], exploited a second-order sliding-mode differentiator to estimate instantaneous indicated torque including friction, and was shown using a Simulink model for a 2 litre I4 gasoline engine, to obtain P_{max} within 5%. A model was combined with a single pressure measurement in [7] to reconstruct pressure on the other cylinders of an I4 DISI engine, showing accuracy within 5 – 10% in P_{max}, and ± 5° on θ_{max}. RBFs trained by recursive hybrid learning of crank or block response data, were used in [13] to predict P_{max} to within 2.9%, and θ_{max} within ± 1.5° on a 9 litre 6-cylinder diesel engine at 39 conditions with speeds varying between 800 rpm and 2000 rpm, from 10% to 90% full load. A NARX ANN was trained in [14] using crank acceleration and fully-recurrent training via the Back-Propagation Through Time (BPTT) algorithm and the Extended Kalman Filter (EKF), to reconstruct P_{max} within 2% for an I3 DISI engine at 1500 rpm and 25.5 Nm. BPTT training was found to be unacceptably slow, and the nominally much faster EKF training was still too slow. Moreover recurrently-trained networks were found to occasionally go seriously unstable. An ANN fed with crank speed (at 1800 rpm), motored-engine pressures, and

spark advance, was tested in [15] using a heat-release model for cylinder pressure, to give results of 5 – 10% accuracy on $P_{max.}$ Crank speed for a four-cylinder Diesel engine was also used in [16] to exploit sliding-mode observer predictions of P_{max} within 2% and θ_{max} within ± 2°. A physical torque model by contrast was used in [17] and tested on a 2-litre 4 cylinder diesel engine at 1500 rpm to produce $P_{max.}$ within the range 2.3% - 11.2% and θ_{max} within -0.4° to 4.4°. Finally an ANN was used in [18] driven by crank speed and acceleration from a single cylinder turbocharged gasoline engine with speeds between 1000 to 2000 rpm to produce P_{max} predictions within 4% - 8%.

Turning attention to indirect cylinder pressure reconstruction using vibration signals the inverse filter concept was extended in [19] for linear system convolution, where Frequency Response Function (FRF) averaging and time-domain Cepstral smoothing was also proposed to reduce the influence of FRF variability, creating a method which they applied to data taken from a single cylinder 4-stroke diesel running full load at 2400 rpm. Vibration signals were processed in [20] using data from a 2-cylinder 4-stroke diesel engine by transforming to the frequency domain and excluding data above 15 kHz then using a Radial Basis Function neural network. Optimal inverse linear filter and averaging using engine condition monitoring principles of ‘cyclostationary processes’ were adopted in [21] to process data from a cylinder head-bolt accelerometer for a 4 cylinder 2-stroke diesel engine running at 900 rpm and various load conditions to produce P_{max} within 5-10%, and a good qualitative match of the pressure trace. Elsewhere measured engine-block vibration was assumed in [22] to comprise three superposed components which appear as filtered versions of the original pressure signals leading to use of decomposition of a time-varying transfer function. This method was used to reconstruct cylinder pressure via four accelerometers, on a 1.8 litre SI engine running at 4000 rpm, giving P_{max} within 25%, and an error mean and standard deviation in θ_{max} of 0.04° and 4.8° respectively. An auto-

regressive-moving-average modelling was used in [23], with Complex Cepstral analysis of high frequency engine-block acoustic emissions, to reconstruct P_{max} within 7% on both a 10000 bhp 2-stroke marine diesel running 25% full load at 77.5 rpm, and a small 4-stroke diesel engine (with speeds ranging from 750 rpm to 1650 rpm). It was suggested however that the same technique would probably not be successful at the lower pressures of a gasoline engine. A feed-forward neural network was applied in [24] to cylinder head vibration data taken from a 6 cylinder 4-stroke diesel engine, where they were able to obtain P_{max} within 2 % and θ_{max} within ±3.5°. A robust (feed-forward) Radial Basis Function ANN was used in [5] to reconstruct cylinder pressure using acceleration signals taken from the cylinder block of a single cylinder diesel engine, enabling P_{max} to be reconstructed within 2.7 % and θ_{max} within ±1.5°, showing a 50 neuron network offered little improvement over 5 neurons.

In this paper recurrent neural network based reconstruction is re-addressed. Previous attempts to train a fully-recurrent network have been severely restricted by the amount of data that can be used owing to the serious inefficiency of the methods, for example in [14]. A faster, more robust, recurrent training method is needed so that more data can be used. Here an adaptive gradient descent method proposed in [25] is applied to gasoline engine cylinder pressure reconstruction. The method combines a weight-update algorithm using standard back propagation (BP) with real-time recurrent learning (RTRL) according to required convergence and stability conditions. The nominal benefit of the method is that the algorithm outperforms both RTRL and normalised RTRL approaches. The objective of the paper is to establish whether these benefits can be realised for crank-kinematic-based gasoline engine cylinder pressure reconstruction to a consistent target accuracy of P_{max} within 4 % and θ_{max} within ±2°.

2. CYLINDER PRESSURE RECONSTRUCTION VIA INVERSE CRANK MODELS

Construction of a dynamic model in the form of a differential equation [26] allows a description of the instantaneous crank motion of an IC engine to be driven by the engine cylinder pressures. This model assumes that all the engine sub-models, and their associated parameters, are known in detail, including the slider-crank mechanism geometry, mass and inertial values, friction properties, and the instantaneous engine load torque. Such models are now routinely used in commercially-available engine simulation codes to predict crank motion but they are not intended to be inverted and used to reconstruct cylinder pressure when supplied with measured kinematics. But there are several reasons why such physical models are not suited to cylinder pressure reconstruction, the main reason arising from uncertainty in the instantaneous engine friction models. Another reason is that the inverse is sensitive to the unknown aspects of various sub-models and their associated parameters.

A clearer way to establish the reasons why reconstructing cylinder pressure via a physical dynamic model is problematic can be approached using the instantaneous crank-acceleration equation:

$$\ddot{\theta} = T_c / I_c \tag{1}$$

where T_c is the instantaneous torque applied to the crank from various sources and I_c is the fixed inertia of the crank. Equation (1) treats the crank as a single rigid body and isolates multiple factors which contribute to the instantaneous torque T_c. These include instantaneous physical effects which are both time and crank angle dependent, namely cylinder pressure forces, reciprocating inertia forces, friction forces, and valve train forces. Cylinder gas pressure is a function of both time, and crank angle, owing to the varying volume of the combustion chamber. The crank-torque arising from gas pressure is crank angle dependent. Cylinder and piston wall temperatures can also influence the magnitude

of the crank-torque from cylinder pressure during transient operation. The crank torque associated with the reciprocating inertia forces, i.e. from the mass of the piston assembly is a function of the crank angle but the relationship does not change for a given engine. Friction forces occur in the main bearings, the big-, and small-end bearings, the piston-ring-pack, and between piston-skirt and liner. The friction torque component is generally crank-angle dependent, but gas pressure affects both the ring-pack contact forces (by acting on the back of the compression rings) and the skirt friction (via the thrust angle with the connecting rod). Auxiliary systems driven by the crankshaft also contribute a frictional component, including the valve-train, the oil, fuel, water, and power-steering pumps, and the alternator. The valve-train component (see [27]) can be sub-divided into friction from the cam bearings, from valve actuation, and from gear, chain, and belt drives, plus valve actuation forces associated with spring loads and valve inertia. The valve torque at the crankshaft is crank angle dependent (being negative during valve opening, and positive during valve closure).

There is an additional difficulty with inversion of a physical model, which is more obvious when dynamic and friction effects are excluded. It is not difficult to show that geometric singularities can prevent local inversion. For example, for a single-cylinder slider-crank mechanism, the instantaneous crank torque T_g for an engine with bore *b*, crank radius *r*, con-rod length *l*, and small *r/l* ratio, is given by:

$$T_g \approx P_g \pi \frac{b^2}{4} \mathrm{r} \sin\theta \left(1 + \frac{r}{l} \cos\theta\right) \qquad (2)$$

where θ is the crank angle and P_g is the instantaneous gas pressure. Inversion of Equation (2) requires instantaneous values T_g and θ to provide an instantaneous value of P_g. It can do this except at two values of θ within a cycle where Equation (2) becomes singular and cannot be evaluated at Top Dead Centre (TDC) when $\theta=0$. This is particularly

problematic as maximum cylinder pressure occurs close to TDC and therefore one of the most important features of the cylinder pressure cannot be accessed via direct physical model inversion. Inversion of a full dynamic model therefore suffers from two problems: i) uncertainty about the sub-models and their associated parameters, and ii) from geometric singularities. Neural network models by contrast offer significant advantages over the inversion of a calibrated physical model as now explained in terms of architecture. To summarise, the purpose of Equation (2) is mainly to create a functional link between crank speed and cylinder pressure excluding uncertainties associated with particular nonlinear effects that arise on a real engine which influence crank kinematics but are not explicit functions of cylinder pressure.

2.1 Cylinder pressure reconstruction via a NARX neural network

The benefit of using a neural network to emulate an inverse crank model is that no prior knowledge of the inverse model structure is needed, and second, the calibration process is generally the same for all classes of network. The challenge is to find the values for the weight sets that provide a correct and robust mapping of the inputs to the outputs. The selection of a network architecture, plus a reliable and efficient weight-training method presents the heart of a successful ANN application. For a given architecture, the ability of a network to successfully predict, is contained in the connection weight matrices.

A feed-forward network architecture is one with a single hidden layer with n inputs, m hidden neurons, and h output neurons. The multiple pathways through the network, combined with non-linear activation functions, provides a very powerful capability for mapping inputs to outputs, and fitting of an arbitrary system model. Training is typically achieved using the standard back-propagation algorithm [28] but input and target vectors need to be pre-processed to avoid a very large range of values. Recurrent Networks are characterised by feedback connections within the network. These feedback connections

can take a wide variety of forms. Feedback paths may be local, i.e. with layer outputs feeding back to the input of the same layer, or global, with values calculated at the output layer of the network on the previous propagation, feeding back to the input layer for the next step. A major difficulty for recurrent network training is that feedback cannot occur until the relevant value in the network has been calculated from the previous propagation.

The Non-linear Autoregressive with Exogenous inputs (NARX) architecture is an important global feedback network (an example shown in figure 1) with the outputs *y(t),* calculated on the preceding step, fed back to be inputs on the next step. The Non-linear title acknowledges the behaviour of the multi-layer perceptron that forms the core of the network, the Autoregressive term acknowledges the fact that the network feeds back its own outputs, and the Exogenous inputs refer to the input vector *u(t)* that is independent of the feedback loops. In addition, both the feedback terms, and the exogenous input vector are shown with additional time delay terms, combining recurrent and time delay architectures. Such recurrent, time-delayed network architectures, can be extremely effective in dynamic system identification because they introduce temporal features to the processing, plus memory and a state representation capability.

The recurrent features of the NARX architecture however create a significant challenge to network training. Although the forward propagation of any of the input vector set through the network, functions in the same way as for the feed-forward architecture described, the next input vector cannot be constructed until the previous prediction is complete, and is then dependent on the weight matrices of the network which are changing during training. Various NARX network training methodologies are available [28], among them Standard Back-Propagation using Teacher Forcing (SBPTF), BPTT, Real Time Recurrent Learning (RTRL), and the use of Kalman Filtering [14][29] to improve the use of the available data.

Here the relatively-new fully-recurrent RAGD Training Algorithm [25] is adopted as now explained.

3. FULLY-RECURRENT TRAINING VIA THE RAGD ALGORITHM

It was shown in [26] and [30] that the recurrent NARX architecture offers great promise for accurate cylinder pressure reconstruction, but robust and efficient training is needed. Both BPTT, and the Extended Kalman Filter (EKF), were shown in [26] and [4] to be too slow. More recently, the Robust Adaptive Gradient Descent (RAGD) algorithm [25] offers promise for efficient NARX training using engine cylinder pressure to create a recurrent network. The guaranteed stability of the RAGD algorithm is attractive and is therefore outlined briefly here, and applied in Section 5.

3.1 The Robust Adaptive Gradient Descent algorithm (RAGD)

The RAGD training algorithm, implemented here for cylinder pressure reconstruction, is a specialisation of the description in [25]. Figure 1 shows the network structure where the overall input vector *x* has *n* elements, comprising: bias, exogenous inputs *u*, each with a value at the current step, and a number of time delays *d.* For cylinder pressure reconstruction, these exogenous inputs can be crank kinematics such as acceleration. The input vector also includes time delays associated with feedback from previous predictions of the output. A (constant) input of unity acts as the 'bias' for all hidden neurons, with the bias-weight trained as an additional column of the hidden weight matrix. The input vector *x,* is of dimension $n \times 1$, here $n = 2 + d + I$. The hidden layer comprises *m* neurons, each with a sigmoid activation function (denoted as $h(\bullet)$) in the form:

$$h(\bullet) = \frac{1}{1 + e^{-\lambda \bullet}} \tag{3}$$

The value λ in equation (3) is implemented as a variable, but remains at unity for all training examples discussed. The hidden layer weight matrix $\hat{W}$ is of dimension $m \times n$, and the output layer has a single linear neuron. The output layer weight vector $\hat{V}$ is of dimension $1 \times n$. Once more, a single constant input value of unity acts as the bias for the output layer, trained as an additional column of the output weight matrix. Here the network output is defined, and during training, the essential steps are given to update the output layer weight vector and the hidden-layer matrix.

The (single neuron) output for (concatenated) cylinder pressure $y(t)=p(t)$, at time t is:

$$y(t) = \hat{V}(t)\,\phi\left(\hat{W}(t)x(t)\right) \tag{4}$$

and is in fact a standard forward propagation evaluation [28]. The vector of hidden-layer output values (the second product in equation (4)) (written as $\phi(t)$ for the remainder of the paper) is evaluated as follows:

$$\begin{aligned}\phi(t) &= \phi\left(\hat{W}(t)x(t)\right) \\ &= \left[h\left(\hat{W}_{1,:}(t)x(t)\right)\ h\left(\hat{W}_{2,:}(t)x(t)\right)\cdots h\left(\hat{W}_{m,:}(t)x(t)\right)\right]^T\end{aligned} \tag{5}$$

During training, the target (desired) output values $d(t)$ are assumed to be corrupted by a disturbance $\varepsilon(t)$. The associated prediction error $e(t)$ is then defined as:

$$e(t) = d(t) - y(t) + \varepsilon(t) \tag{6}$$

and the instantaneous cost function associated with the RAGD algorithm is taken as:

$$E(t) = \frac{e^2(t)}{2} \tag{7}$$

The training objective is to update the output layer weight vector $\hat{V}$ and the hidden layer weight matrices $\hat{W}$ such that $E(t)$ is minimised.

3.2 Output-Layer Weight matrix update

The RAGD algorithm uses an adaptive hybrid learning algorithm, working in both standard online Back-Propagation (BP) and Real Time Recurrent Learning (RTRL) according to stability and convergence conditions. The output weight update equation is given as:

$$\hat{V}(t+1) = \hat{V}(t) + \frac{\alpha^{v}(t)}{\rho^{v}(t)} \cdot e(t) \cdot \left[\phi(t)^{T} + \beta^{v}(t) \cdot \hat{A}(t) \right] \tag{8}$$

where $\alpha^{v}(t)$ is an adaptive learning rate defined as:

$$\begin{aligned} \alpha^{v}(t) &= 1 \qquad \text{if} \quad |e(t)| \geq \varepsilon_{m}^{v} \Big/ \sqrt{1 - \frac{\left\| \phi(t)^{T} + \beta^{v}(t) \hat{A}(t) \right\|^{2}}{\rho^{v}(t)}} \\ \alpha^{v}(t) &= 0 \qquad \text{if} \quad |e(t)| < \varepsilon_{m}^{v} \Big/ \sqrt{1 - \frac{\left\| \phi(t)^{T} + \beta^{v}(t) \hat{A}(t) \right\|^{2}}{\rho^{v}(t)}} \end{aligned} \tag{9}$$

and where $\varepsilon_{m}^{v} = \max\left(\left| \tilde{\varepsilon}^{v}(t) \right| \right)$, and $\rho^{v}(t)$ is a normalisation factor such that:

$$\rho^{v}(t) = v \rho^{v}(t-1) + \max\left\{ \bar{\rho}^{v}, \left\| \phi(t)^{T} + \beta^{v}(t) \hat{A}(t) \right\|^{2} \right\} \tag{10}$$

with $0 < v < 1$ and $0 < \bar{\rho}^{v}$ being positive constants, and where the variable $\beta^{v}(t)$ in equations (8) - (10)), is a hybrid adaptive learning rate defined by:

$$\begin{aligned} \beta^{v}(t) &= 1, \qquad \text{if} \quad \phi^{T}(t) \left\{ \delta I + \phi(t) \phi^{T}(t) \right\}^{-1} \hat{A}^{T}(t) \geq 0 \\ \beta^{v}(t) &= 0, \qquad \text{if} \quad \phi^{T}(t) \left\{ \delta I + \phi(t) \phi^{T}(t) \right\}^{-1} \hat{A}^{T}(t) < 0 \end{aligned} \tag{11}$$

with δI a small positive constant; $\hat{A}(t)$ is a 1 x *m* extended recurrent gradient of the form:

$$\hat{A}(t) = \hat{V}(t) \phi'(t) \hat{W}(t) \hat{D}_{v}(t) \tag{12}$$

where $\hat{D}_{v}(t)$ is the *n x m* Jacobian matrix defined as:

$$\hat{D}_v(t) = \left[\frac{\partial u(t)}{\partial \hat{V}(t)} \cdots \frac{\partial u(t-d)}{\partial \hat{V}(t)} \; \frac{\partial y(t-1)}{\partial \hat{V}(t)} \cdots \frac{\partial y(t-l)}{\partial \hat{V}(t)} \right] \approx \left[\frac{du(t)}{d\hat{V}(t)} \cdots \frac{du(t-d)}{d\hat{V}(t)} \; \frac{dy(t-1)}{d\hat{V}(t-1)} \cdots \frac{dy(t-l)}{d\hat{V}(t-l)} \right] \tag{13}$$

and $\phi'(t)$ is an *m x m* diagonal matrix given by:

$$\phi'(t) = diag\left[\phi_1'(t) \; \phi_2'(t) \cdots \phi_m'(t) \right] \tag{14}$$

Normally the exogenous inputs and the bias do not change with the weight matrix, so the first $d+2$ entries of $\hat{D}_v(t)$ are zero vectors. Equations (9) - (14) provide the essentials needed to evaluate equation (8) such that the output layer weight matrix can be updated.

3.3 Hidden-Layer Weight matrix update

The hidden layer weight matrix update uses an adaptive normalised gradient algorithm constructed in a similar manner to that for the output layer, although some aspects are more complicated as there are many hidden neurons compared to a single output neuron. The hidden layer weight update is given as:

$$\hat{W}(t+1) = \hat{W}(t) + \frac{\alpha^w(t)}{\rho^w(t)} \cdot e(t) \cdot \left[\phi'(t) \hat{V}^T(t) x^T(t) + \beta^w(t) \hat{B}(t) \right] \tag{15}$$

where $\alpha^w(t)$ is an adaptive learning rate defined as:

$$\alpha^w(t) = 1 \;\; if \;\; |e(t)| \geq \varepsilon_m^w \Big/ \sqrt{1 - \frac{\left\| \phi'(t) \hat{V}^T(t) x^T(t) + \beta^w(t) \hat{B}(t) \right\|^2 \Big/ h'_{\min}(t)}{\rho^w(t)}}$$
$$\alpha^w(t) = 0 \;\; if \;\; |e(t)| < \varepsilon_m^w \Big/ \sqrt{1 - \frac{\left\| \phi'(t) \hat{V}^T(t) x^T(t) + \beta^w(t) \hat{B}(t) \right\|^2 \Big/ h'_{\min}(t)}{\rho^w(t)}} \tag{16}$$

and where $\varepsilon_m^v = \max\left(\left| \tilde{\varepsilon}^v(t) \right| \right)$ and $\rho^w(t)$ is a normalisation factor such that:

$$\rho^w(t) = v\rho^w(t-1) + \max\left\{\bar{\rho}^w, \left\|\phi'(t)\hat{V}^T(t)x^T(t) + \beta^w(t)\hat{B}(t)\right\|^2 \Big/ h'_{\min}(t)\right\} \tag{17}$$

with $0 < v < 1$ and $0 < \bar{\rho}^w$ again being positive constants, and:

$$h'_{\min}(t) = \min\left\{\phi'_1(t)\ \ \phi'_2(t)\ \cdots\ \phi'_m(t)\right\} \neq 0 \tag{18}$$

The parameter $\beta^v(t)$ is a hybrid adaptive learning rate defined by:

$$\begin{aligned} &\beta^w(t) = 1, \quad \text{if } \underline{\hat{W}}(t)\hat{D}_w(t)\left\{\delta I + x(t)x^T(t)\right\}^{-1}x(t) \geq 0 \\ &\beta^w(t) = 0, \quad \text{if } \underline{\hat{W}}(t)\hat{D}_w(t)\left\{\delta I + x(t)x^T(t)\right\}^{-1}x(t) < 0 \end{aligned} \tag{19}$$

with δI a small positive constant; $\hat{B}(t)$ is the extended *m x n* recurrent gradient of the defined as:

$$\hat{B}(t) = \phi'(t)\hat{V}^T(t)\underline{\hat{W}}(t)\hat{D}_w(t) \tag{20}$$

where $\underline{\hat{W}(t)}$ is a long vector version of the *m x n* hidden layer weight matrix $\hat{W}(t)$ i.e.:

$$\underline{\hat{W}(t)} = \left[\hat{W}_{1,:}(t)\ \ \hat{W}_{2,:}(t)\ \cdots\ \hat{W}_{m,:}(t)\ \right] \tag{21}$$

and $\hat{D}_w(t)$ is the Jacobian matrix for the hidden layer defined as:

$$\hat{D}_w(t) = \left[\hat{D}_w^1(t)\ \ \hat{D}_w^2(t)\ \cdots\ \hat{D}_w^m(t)\right] \tag{22}$$

where:

$$\begin{aligned} \hat{D}_w^i(t) &= \left[\ \frac{\partial u(t)}{\partial \hat{W}_{i,:}(t)}\ \cdots\ \frac{\partial u(t-d)}{\partial \hat{W}_{i,:}(t)}\ \frac{\partial y(t-1)}{\partial \hat{W}_{i,:}(t)}\ \cdots\ \frac{\partial y(t-l)}{\partial \hat{W}_{i,:}(t)}\ \right]^T \\ &\approx \left[\ \frac{du(t)}{d\hat{W}_{i,:}(t)}\cdots\frac{du(t-d)}{d\hat{W}_{i,:}(t)}\ \frac{dy(t-1)}{d\hat{W}_{i,:}(t-1)}\cdots\frac{dy(t-l)}{d\hat{W}_{i,:}(t-l)}\ \right]^T \end{aligned} \tag{23}$$

Equations (16) - (23) provide the essential steps required to evaluate equation (15) to update the hidden layer weight matrix.

4. ENGINE MEASUREMENTS, DATA ACQUISITION AND PROCESSING

High quality engine test data is a pre-requisite for neural network training. This section outlines the engine data, the data acquisition hardware, and the signal processing methods to make the data suitable for application to ANN training. Action needed to be taken to overcome crank angle encoder inaccuracies when used outside its intended design function to generate crank kinematics, along with ways of synchronising acquisition of time-based and crank angle-based data. A summary of the problem and the solution is given shortly - full details are available in [30].

Data for input to the ANN training was obtained from a 1.125 litre 3-cylinder 4-stroke (Ford) inline DISI engine connected directly via flexible coupling to a McClure 130kW/7000 rev/min DC dynamometer. Figure 2 shows the layout of the engine measurement system. The first driveline system natural frequency in torsion being 16.5 Hz. Cylinder pressures were measured on all 3 cylinders using Kistler type 6117BCD36 spark plug integrated transducers (with a range 0-150 bar) and Kistler Type 5044 charge amplifiers. Crank angle displacement was measured at the crank nose using a Kistler Type 2614A1 optical encoder, with signal conditioning through a Kistler Type 2614A4 pulse multiplier.

The data acquisition system was based on National Instruments (NI) hardware, controlled and programmed using LabVIEW software. The data acquisition hardware was an NI PXI system comprising PXI-8331 Interface to Windows PC, PXI-6133 Analogue input module with 14 bit synchronous sampling across 8 channels using a TB-2709 terminal block for low noise co-axial cable connection with a maximum sample rate of 2.5MS/s, and a maximum input amplitude of 10V, and PXI-6602 Counter/timer with 32-bit counters and 80MHz maximum source frequency, using a BNC-2121 terminal block for low noise co-axial cable connections

The shaft encoder offers 2 TTL signal output streams: 1 pulse-per-revolution (ppr), and optionally either 360 ppr or 3600 ppr. The encoder was carefully aligned such that the rising edge of the 1 pulse-per-revolution was coincident with piston top dead centre (TDC). The 2nd output pulse trained was configured for 360 ppr (i.e. 1° crank rotation pulse spacing). The 3600 ppr option was actually found to be unsuitable for processing crank kinematics because although the rising edges of the 1 ppr output (at TDC) and the first 1° marker were coincident, there was limited accuracy of the 1° pulses. Both pulse trains are connected to 80 MHz counter channels on the PXI-6602. The output from the counter channel provides high time resolution durations between consecutive rising edges for each pulse train. The reciprocal of duration between rising edges gives mean rotation speed – per revolution for the 1ppr signal and per degree for the 360 ppr signal. Use of central finite difference approximation to numerically differentiate the 360 ppr velocity estimate provides crank shaft acceleration.

With the main use of the crank encoder to derive crankshaft acceleration, small (manufacturing) variability evident in the pulses nominally associated with 1°, produced high levels of noise after twice numerical differentiation. A novel method of calibrating the encoder and subsequently filtering the noise to minimise the effects of this issue had to be developed. This method is now summarised.

Crank encoder calibration

Direct processing of raw data using the Kistler crank encoder to obtain crank velocity and acceleration reveals a serious problem. Velocity data shows high levels of noise, which are ultimately found to be repeatable across every revolution. The magnification of these noisy components by differentiating the velocity signal to obtain acceleration produces totally unacceptable results. Crank velocity measurements under engine motored conditions at 1000 rev/min for example, show large low frequency variations of around ±

20 rev/min in velocity which are the expected result of piston deceleration and acceleration. There are however significant additional high frequency variations, which following differentiation, make the in-cylinder pressure induced accelerations indistinguishable from the higher frequency components. This is evidence of corruption but the pattern of the high frequency noise is repeated over every revolution, which is revealed by overlaying successive engine cycles. The cause of the problem stems from the degree markers on the encoder not being manufactured with sufficiently accurate spacing to allow crank acceleration to be computed. Every degree pulse appears to have a repeatable error, and as the encoder completes 360° the cumulative angular error is effectively reset. The encoder, which is of industry standard, is used widely for combustion analysis. In that role, it is used to indicate crank position rather than to obtain velocity and acceleration. Small errors in each angle are not then a problem. But when differentiated, the error magnitude increases dramatically.

To solve the problem, an encoder calibration procedure has been developed. An initial attempt was to use 'constant speed' calibration. By turning the encoder at constant speed, the degree pulse 'rising edges' should be equally-spaced in time. Any deviation from equal-time spacing could be measured, and a correction calculated. This procedure effectively identifies (to a high resolution) how many degrees a pulse actually represents rather than assuming precisely 1° for each pulse. By accurately measuring the time interval between TDC positions on a disc (i.e. 360° CA) involving either constant, or slowly-varying speed ω, the data could be used to fit a simple polynomial (such as a cubic-spline) for interpolation purposes, which would allow the time intervals t_{int} between ideal 1° crank angles, to be interpolated. Designating the measured time interval between the actual TTL pulses as t_{TTL} the error ε_i can then be estimated for i=1°,…,360° where $\varepsilon_i = \omega_i(t_{\text{int}} - t_{TTL})$. Achieving a perfectly constant rotation speed is however not

straightforward because most electric motors have some torque-variation-induced speed ripple per revolution. Therefore driving the encoder exactly at constant speed is not possible. Instead, when a disc of known 2nd moment of mass is attached to the encoder drive face, sufficient energy is stored to enable the encoder to coast down to rest in around 15 revolutions, under internal friction only. A steel disk of inertia 475×10^{-6} kgm^2 for example, exhibits precisely this behaviour. The internal friction was assumed to vary as a smooth function of rotational speed but not with each complete revolution. To measure the disc coasting down to rest, both the pulse trains corresponding to the 1 ppr and the 360 ppr TTL signals were captured with the NI data acquisition system using the 80MHz counters to set the sampling rate. The 1 ppr signal was assumed to have no angular displacement error – where the pulses must be physically 360° apart as they are generated by the same optical slot in the encoder disc. A cubic spline fit polynomial is then used to model the time at which 1° spaced pulses should appear. The difference between these modelled times and the 360 ppr TTL signal times, measured by the counter, gives the encoder error that must be re-calibrated. Regarding low pass filtering of the calibrated encoder data to reduce the noise associated with twice differentiating a measured signal, a low pass Chebyshev Type 2 IIR filter was used, with 8 poles and 30 dB stopband attenuation. The cut-off frequency for each set of data was set at the 18th order of crank rotation, i.e. 300 Hz at 1000 rpm, 450 Hz at 1500 rpm, and 600 Hz at 2000 rpm. In each case, the filter was implemented using the Matlab '*filtfilt*' function to achieve zero phase distortion by filtering in both forward and reverse directions.

Repeated revolutions give very similar results, and a cyclic mean over 10 revolutions is used as the set of final calibration values. When the calibrated angles for each of the 360 ppr pulses have been derived, they are then used to calculate the angular velocity of the crank instead of taking the nominal 1° values. The calibrated values result in significantly

smoother velocity histories, with much-reduced sample-to-sample variation. This also eliminates the need for a large correction at 0° encoder angle i.e. at TDC for Cylinder-1. Moreover numerically differentiating the calibrated velocity signal to give crank acceleration, gives a very significant improvement in the signal-to-noise ratio of the resulting acceleration data. However, the acceleration signal remains somewhat noisy, which is a typical consequence of numerically differentiating any measured time history. The full justification and benefits of this encoder calibration procedure for obtaining crank kinematics, are given in [30]. Figure 3 shows an example of measured crank acceleration derived from calibrated encoder data demonstrating the effect of noise reduction via changes in cut-off frequency.

A somewhat different data acquisition problem was that constant crank-angle-clocked data had an effective sampling frequency which varies with engine speed: i.e. at 1000 rev/min, a 1° sampling interval results in a 6 kHz sample rate (6000 rev/min equates to 36 kHz). Using the analogue input card internal time-base, samples were clocked up to 2.5 MHz, avoiding aliasing issues. To ensure synchronisation between captured analogue signals and the crank kinematics, the TDC pulse from the encoder was used as a common trigger to initiate acquisition on all input channels simultaneously.

4.1 Cylinder pressure measurement

Accurate cylinder pressure is an obvious requirement of the test data. If this data is incorrectly scaled or noisy, or drifts through the test recording, or is not consistent across cylinders, then any attempt to use these pressures as training or validation data for an ANN is flawed from the outset. To achieve the best possible dataset, each transducer/ charge-amplifier pair was statically calibrated prior to testing, using a Druck DPI603. The charge amplifiers were reset immediately prior to each data recording. However, the very nature of the piezo-electric sensors and charge amplifiers means that referencing the

output to a known pressure is necessary (pegging). Setting the pressure equal to inlet manifold pressure at Bottom Dead Centre (BDC) when the inlet valves are open, is acceptable for low-speed/low-load conditions where manifold tuning effects are small [31]. This 'per-cycle' approach was modified by correcting the errors in the manifold pressure at two consecutive BDC points for a 720° trace. This approach avoids the potential discontinuity in the corrected pressure trace that would otherwise occur between cycles.

To establish appropriate data acquisition rates, signal processing bandwidths, and suitable filtering of noisy signals, while preserving important frequency ranges, two measures were adopted. Fourier analysis was used to study the amplitudes of the time averaged frequency content over 1 pressure cycle, and low-pass filtering was used to understand the influence of bandwidth on peak pressure magnitude P_{max}, and position θ_{max}. Figure 4 shows the effect of low-pass filtering with various cut-off frequencies, showing that cutting-off below 1000 Hz will cause errors in the filtered pressure signal.

When training an inverse ANN model to reconstruct cylinder pressure from crank kinematic data an issue arises for time-domain-based reconstruction of multi-cylinder engine pressures that is not present for just one cylinder. The crank acceleration is driven by torque from 3 cylinders. If cylinder pressure is to be reconstructed for one specific cylinder, then the ANN must create an inverse model which will receive a single input with 3 events per cycle. It is then required to generate an output with a single event per cycle. If the reconstruction is to apply for individual cylinders, then the network must be able to separate events within the cycles. Network training is simplified if the cylinder pressure from three cylinders is combined into a single trace, such that the ANN structure would then have a single output to reconstruct from a single input signal. The approach used to generate such a single pressure signal is straightforward, as an example shows in figure 5. Three individual traces are shown for 1 engine cycle in figure 5 (top); the combined

pressure trace by contrast (which is used for ANN training) is shown in figure 5 (bottom). The most important features of the pressure trace around TDC firing for each cylinder, are not affected by this type of data merging. Measured cylinder pressure data under various load conditions has been captured from the 3-cylinder DISI engine discussed in Section 4. When processed to show the cycle-by-cycle variability in peak pressure and location of peak pressure across all three cylinders, it is found [30] that peak pressure, and the crank angle location of peak pressure exhibit considerable variability for most conditions of operation – the most variable being low speed and low load (the least variable being high speed and low load). This general level of variability presents a considerable challenge for reliable reconstruction of cylinder pressure using an inverse crank dynamic model. This will now be examined using a RAGD-trained NARX model.

5. CYLINDER PRESSURE RECONSTRUCTION VIA RAGD-TRAINED NARX MODEL

The RAGD algorithm of Section 3 is now applied to measured engine data, obtained using the facilities, data acquisition methodology, and processing procedure described in Section 4, to train the recurrent NARX network shown in figure 1. First, rather than apply the RAGD algorithm directly to measured data, the network properties and the RAGD algorithm training parameters are tuned using synthesised crank kinematics obtained using measured cylinder pressures and a very much simplified crank dynamic model. After tuning the RAGD algorithm on synthetic crank acceleration data, attention then turns to train a NARX network using measured engine data i.e. using both measured crank kinematics and measured cylinder pressures. The effectiveness of NARX network reconstruction is reported for two types of kinematic input i.e.: a network with a single input comprising measured crank acceleration, followed by a network with two inputs comprising measured crank velocity and acceleration.

5.1 Tuning of the RAGD training algorithm using synthesised crank acceleration

To assist in tuning the RAGD training algorithm synthesised crank acceleration [30] was generated via equation (2) to relate measured cylinder pressure to crank acceleration. Torque histories were calculated from equation (2) for each of the 3 cylinders by appropriate phasing, and then summed. This combined torque was then divided by the total inertia of the cranktrain to produce a synthesised crank acceleration as follows:

$$\ddot{\theta} = \frac{\pi b^2 r \sin(\theta)\left\{1 + \frac{r}{l}\cos(\theta)\right\} P_{\Sigma g}}{4 I_c} \tag{24}$$

where $P_{\Sigma g}$ is the combined (measured) cylinder pressure trace, I_c is the crank-train inertia, *b*, *r*, and *l* are respectively: the cylinder bore, crank radius, and con-rod length, θ is the crank angle from TDC, and $\ddot{\theta}$ is the crank acceleration. The use of synthesised crank acceleration avoids any risk of the training algorithm being sensitive to the data processing problems discussed in Section 4. The resulting synthesised crank acceleration, is used as the exogenous input to the NARX network. Using synthesised crank kinematics, the data requirements of the RAGD algorithm can be established, in particular, several important learning rates. Also the NARX network architecture can be optimised in terms of the required number of hidden-layer neurons. Figure 6 shows an example of synthesised crank acceleration obtained from measured i3 DISI engine cylinder pressure (as described in Section 4).

Regarding the data requirements, experience of NARX network training using the RAGD algorithm reveals that use of the final set of (time-dependent) weights does not produce accurate predictions. In fact, two issues became clear: i) a very large number of engine cycles are needed to fully train the network (for example 195 engine cycles proves totally insufficient to achieve stable training), and ii) the rate at which the weights should be changed needs to be carefully restricted. This is to reduce sample-by-sample tracking of

the target, i.e. so that the weights would develop slowly across several cycles rather than rapidly within individual cycles. Appropriate weight-update-rates were achieved by selecting suitable values for the constants: v, $\bar{\rho}^{v}$, and $\bar{\rho}^{w}$ appearing in equations (10) and (17). These constants influence the magnitudes of $\rho^{v}(t)$ and $\rho^{w}(t)$ which govern the weight-update-rates for both the hidden-layer and output-layer weight matrices. The higher the values for these constants, the lower the respective update rates. Each constant was eventually tuned to the same value = 0.9.

The problem associated with the need for large training data sets (i.e. to achieve stable convergence for the weight values) was circumvented by structuring the training routine to re-use data sets. For example, a set of cycles was presented, producing weights that were updated as each sample was presented. Then at the end of a set of cycles (an 'epoch'), the data set was re-presented to the algorithm, by setting the initial weights to the final weight values from the previous presentation. Similarly, the recurrent inputs at the start of an epoch were those obtained from the previous epoch. Three other RAGD parameters required knowledge of their values at the previous time step, namely: i) $\rho^{v}(t-1)$, ii) $\rho^{w}(t-1)$, and iii) the recurrent terms in the Jacobian matrices. Finally, at the end of each epoch, the time-averaged weight-values over that particular epoch were computed, and used as a set of static weights for a NARX prediction.

To tune the RAGD training algorithm, initially a single nominal engine test point was examined, namely at 1000 rev/min and 10 Nm test, and a series of training iterations was undertaken using the RAGD algorithm with a varying numbers of hidden-layer neurons range from 4 to 24. These training iterations used 10 engine cycles per epoch, because this number provided sufficient data (i.e. 30 pressure peak events) to give reasonable training time per epoch to allow progress to be monitored. Training was allowed to run for a total of 500 epochs. In terms of the network architecture, NARX networks with the

structure shown in figure 1, were constructed with 6 exogenous delayed inputs (i.e. synthesised crank acceleration), and 6 time-step feedback delays from the network output forming the fully recurrent structure.

Figure 7 shows the errors in predicted peak pressure and location of peak pressure as a function of the number of epochs, using synthesised crank acceleration as input data for a network with 8 hidden-layer neurons at nominal engine test point: 1000 rev/min, 10 Nm. Figure 8 shows the corresponding errors in predicted peak pressure and location of peak pressure at the same test point when the network has 24 hidden-layer neurons. To enable errors less than 20% to be clearly visible on a linear scale, errors above 20% in Figures 7 and 8 (and indeed in subsequent error figures) are shown saturated at 20%. Comparing figures 7 and 8 it is clear that training a network with just 8 hidden-layer neurons is problematic whereas a network with 24 hidden-layer neurons is showing early convergence of the static weights. Moreover it should be stated that the errors are minimised long before the total number of epochs has been reached.

Figure 9 shows predicted cylinder pressure traces using synthesised crank acceleration corresponding to both training and unseen (generalisation) data using the static weights corresponding to the 47^{th} training epoch of the NARX network with 24 hidden-layer neurons. Figure 10 shows the corresponding errors in predicted peak pressure, and location of peak pressure, for RAGD training using the 47^{th} training epoch. These results show that a NARX network tuned using synthesised crank acceleration produce stable training characteristics and good generalisation capability at the same engine test point. To examine the capability of a network trained at one nominal engine test point to predict cylinder pressures at a different test point, two such cases were considered: 1000 rev/min and 20 Nm; and 1500 rev/min, 10 Nm, i.e. the same speed but different load, then different speed but the same load. Table 1 summarises the error results. It is clear from the results

in Table 1 that a RAGD trained NARX network, using the tuning methodology adopted, that predictions at the same speed but different load are reasonable but peak pressure predictions at a different speed are poor. The question that remains is how well at the same test point will a RAGD trained NARX network predict pressure traces using real engine data.

Table 1 – Peak pressure error statistics for a NARX model predicting unseen data

Predictions using Synthesised Crank Acceleration at engine test points	**Network trained at 1000 rev/min, 10 Nm**			
	Mean error P_{max} [%]	2σ Error P_{max} [%]	Mean error θ_{max} [^{o}]	2σ error θ_{max} [^{o}]
1000 rev/min, 10 Nm	5.3	9.1	1.7	3.4
1000 rev/min, 20 Nm	7.7	12.9	3.8	4.8
1500 rev/min, 10 Nm	33.6	10.3	4.3	1.4

5.2 Cylinder pressure reconstruction using measured engine data

To establish how well a RAGD trained NARX network will predict pressure traces at the same test point using real engine data, a network with a single input comprising measured crank acceleration is initially examined, followed immediately by examination of a network comprising two inputs i.e. measured crank velocity and crank acceleration.

Using the same engine test point i.e.: 1000 rev/min and 10 Nm, and a RAGD-trained network with 24 hidden-layer neurons, figure 11 shows the training errors in predicted peak pressure $P_{max,}$, and location of peak pressure θ_{max}, as a function of the number of epochs, using measured crank acceleration as input data. Figure 12 shows the predicted cylinder pressure using measured crank acceleration corresponding to both training data and unseen (generalisation) data using the static weights of the 82nd training epoch of the NARX network. Figure 13 shows the errors associated with the predicted peak pressure,

and location of peak pressure for RAGD training via measured crank acceleration using the static weights of the 82nd training epoch.

A study using crank velocity as the input (rather than acceleration) proved less accurate but actually gave more robust and faster training suggesting possible merit in using two inputs at the same test point. Figure 14 shows, using measured crank velocity and acceleration, the training errors in predicted peak pressure and location of peak pressure as a function of the number of epochs, using 10 engine cycles. Figure 15 shows the corresponding predicted cylinder pressure for unseen (generalisation) data using the static weights of the 27th training epoch of a NARX network, and figure 16 shows the associated errors in predicted peak pressure, and location of peak pressure.

The results of tuning the RAGD algorithm for training a NARX model shows that a network used to predict at the same engine test point as trained on, reliably predicts. As figure 12 shows, unseen engine cylinder pressures using synthesised crank acceleration are particularly accurate, typically within 4% for $P_{max,}$ and ± 2° for the location of peak pressure θ_{max}. When real engine data is tested using measured crank acceleration, the error in $P_{max,}$ is higher, typically within 10% for P_{max}, whereas the error in θ_{max} is consistently around ± 2°. However when engine data corresponding to a different test point is used, the results deteriorate, especially at difference engine speeds. Figure 16 shows that when using both engine crank velocity and crank acceleration as input data, training is more stable and faster (than when crank acceleration or crank velocity are used separately). The accuracy associated with $P_{max,}$ and θ_{max} is actually very similar to the use of crank acceleration alone making the combination of crank acceleration and crank velocity as input information, the most practical NARX architecture to use.

6. CONCLUSIONS

A fully-recurrent training algorithm has been adapted to train NARX neural networks to predict engine cylinder pressure traces using crank kinematics as input data. Previously, no suitably robust and efficient training method has been available. Neural networks are shown to be more appropriate than physical crank dynamic models, which when inverted, suffer from singular behaviour where accuracy is most needed. The proposed NARX architecture is trained using measured engine crank kinematics obtained from an industry-standard shaft encoder which requires special processing actions to render the data useful. The training algorithm i.e. the Robust Adaptive Gradient Descent (RAGD) method, is first tuned and tested using synthesised crank acceleration data to avoid any risk that the special data processing would undermine network performance. The paper shows that a NARX network trained using a tuned RAGD algorithm and measured engine crank acceleration, produces results within 10% for $P_{max,}$ and within $\pm 2°$ for the location of peak pressure θ_{max}. The same network is however not accurate at different test points, particularly different engine speeds. But it is also shown that a RAGD-trained NARX network using both crank velocity and crank acceleration, is just as accurate as for crank acceleration alone but training is much faster and more robust making it the most practical NARX architecture and recurrent training methodology to use on production engines.

Acknowledgements

The authors wish to acknowledge funding support for this project from both the EPSRC and Jaguar Land Rover, under Contract Number: EP/E03246X/1. The considerable technical support is acknowledged of colleagues at Powertrain Research & Technology - Jaguar Land Rover, Coventry, and earlier, at Jaguar Land Rover - Advanced Powertrain Engineering, Whitley Engineering Centre, Coventry.

References

[1] Müller R., Hart M., Krötz G., Eickhoff M., Truscott A., Noble A., Cavalloni C., Gnielka M. (2000), Combustion Pressure Based Engine Management System. SAE Technical Paper 2000-01-0928.

[2] Yoon P., Park S., Sunwoo M., Ohm I., Yoon K.J. (2000), Closed-loop control of spark advance and air-fuel ratio in SI engines using cylinder pressure. SAE Technical Paper 2000-01-0933.

[3] Park S., Sunwoo M. (2003), Torque estimation of spark ignition engines via cylinder pressure measurement. Proceedings of the Institute of Mechanical Engineers Part D: Journal of Automobile Engineering September 2003 Vol. 217 p809-817.

[4] Vulli S. (2006), Engine cylinder pressure reconstruction using neural networks and knock sensor measurements. Doctoral thesis, University of Sussex, 2006.

[5] Bizon K., Continillo G., Mancaruso E., Vaglieco B.M. (2011), Reconstruction of in-cylinder pressure in a diesel engine from vibration signal using a RBF neural network model. SAE Technical Paper 2011-24-0161.

[6] Potenza R. (2006), Engine cylinder pressure reconstruction using neural networks and crank kinematics. Doctoral thesis, University of Sussex 2006.

[7] Hamedovi H., Raichle F., Bohme J.F. (2005), In-cylinder pressure reconstruction for multi-cylinder SI engine by combined processing of engine speed and one cylinder pressure. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) March 2005, Proceedings Volume 5, p677-680.

[8] Lee B., Guezennec Y.G., Rizzoni G. (2001), Estimation of cycle-resolved in-cylinder pressure and air-fuel ratio using spark plug ionization current sensing. International Journal of Engine Research 2001 Vol. 2 No. 4 p263-276.

[9] Rizzoni G. (1989), Estimate of indicated torque from crankshaft speed fluctuations: A model for the dynamics of the IC engine. IEEE Transactions on Vehicular Technology, Vol. 38, Issue 3, August 1989.

[10] Jacob P.J., Gu F., Ball A.D. (1999), Non-parametric models in the monitoring of engine performance and condition Part 1: Modelling of non-linear engine processes. Proceedings of the Institution of Mechanical Engineers Part D 1999, Vol.213(1) p73-81.

[11] Gu F., Jacob P.J., Ball A.D. (1999), Non-parametric models in the monitoring of engine performance and condition Part 2: Non-intrusive estimation of diesel engine cylinder pressure and its use in fault detection. Proceedings of the Institution of Mechanical Engineers Part D 1999, Vol 213(2) p135-143.

[12] Haskara I., Mianzo L. (2001), Real-time cylinder pressure and indicated torque estimation via second order sliding modes. Proceedings of the American Control Conference, 2001 Vol. 5 p3324-3328.

[13] Johnsson R. (2006), Cylinder pressure reconstruction based on complex radial basis function networks from vibration and speed signals. Mechanical Systems and Signal Processing Vol. 20 Issue 8 Nov. 2006 p1923-1940.

[14] Potenza R., Dunne J.F., Vulli S., Richardson D., King P. (2007), Multicylinder engine pressure reconstruction using NARX neural networks and crank kinematics. International Journal of Engine Research 2007 Vol. 8 No. 6 p499-518.

[15] Saraswati S., Chand S. (2010), Reconstruction of cylinder pressure for SI engine using recurrent neural network. Neural Computing & Applications September 2010 Vol. 19 Issue 6 p935-944.

[16] Al-Durra A., Fiorentini L., Canova M., Yurkovich S. (2011), A model-based estimator of engine cylinder pressure imbalance for combustion feedback control applications. American Control Conference 2011 p991-996 ISSN:0743-1619.

[17] Liu F., Amaratunga A.J., Collings N., Soliman A. (2012), An experimental study on engine dynamics model based in-cylinder pressure estimation. SAE Technical Paper 2012-01-0896.

[18] Taglialatela F., Lavorgna M., Mancaruso E., Vaglieco B.M. (2013), Determination of combustion parameters using engine crankshaft speed. Mechanical Systems and Signal Processing Vol. 38 Issue 2 p628-633.

[19] Gao Y., Randall R.B. (1999), Reconstruction of diesel engine cylinder pressure using a time domain smoothing technique. Mechanical Systems and Signal Processing Vol.13, Issue 5 September 1999 p709-722.

[20] Du H., L Zhang L., Shi X. (2001), Reconstructing cylinder pressure from vibration signals based on radial basis function networks, Proceedings of the Institution of Mechanical Engineers Part D 2001. Vol. 215(6) p761-767.

[21] Antoni J., Daniere J., Guillet F., Randal R.B. (2002), Effective vibration analysis of IC engines using cyclostationarity Part ii} New results on the reconstruction of the cylinder pressures. Journal of Sound and Vibration Volume 257 Issue 5 November 2002, p839-856.

[22] Villarino R., Böhme J.F. (2003), Fast in-cylinder pressure reconstruction from structure-borne sound using the EM algorithm. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) April 2003 Vol. 6 p597-600.

[23] El-Ghamry M., Steel J.A., Reuben R.L., Fog T.L. (2005), Indirect measurement of cylinder pressure from diesel engines using acoustic emission. Mechanical Systems and Signal Processing Vol.19, Issue 4 July 2005 p751-765.

[24] Yong X., Guiyou H., Chunrong S., Zhibing N., Wu Z. (2010), Reconstruction of cylinder pressure of I.C. engine based on neural networks. First International Conference on Pervasive Computing, Signal Processing and Applications, September 2010 p924-927.

[25] Song Q., Wu Y., Soh Y.C. (2008), Robust adaptive gradient descent training algorithm for recurrent neural networks in discrete time domain. IEEE Transaction on Neural Networks November 2008 Vol. 19, No.11, p1841-1853.

[26] R. Potenza, J. F. Dunne, S. Vulli and D. Richardson. (2007) A Model for Simulating the Instantaneous Crank Kinematics and Total Mechanical Losses in a Multi-Cylinder In-Line Engine. *International Journal of Engine Research*. 8(4), 379-397.

[27] Mufti R.A., Priest M. (2012), Effect of cylinder pressure on engine valve-train friction under motored and fired conditions. Proceedings of the Institution of Mechanical Engineers, Part J: Engineering Tribology April 2012 Vol. 226 No. 4 p306-314.

[28] Haykin S. (1999), Neural networks a comprehensive foundation (Second edition). Prentice Hall International Inc. 1999 ISBN 0139083855.

[29] Haykin, S (Ed.) (2001) Kalman Filtering and Neural Networks. Wiley-Interscience,

[30] Bennett C. (2014) ‘Reconstruction of Gasoline Engine In-Cylinder Pressures Using Recurrent Neural Networks’, PhD Thesis, University of Sussex.

[31] Lee K., Yoon M., Sunwoo M. (2007), A study on pegging methods for noisy cylinder pressure signal. *Control Engineering Practice* 2008, Vol. 16 Issue 8, 922-929.

List of Figures

FIGURES

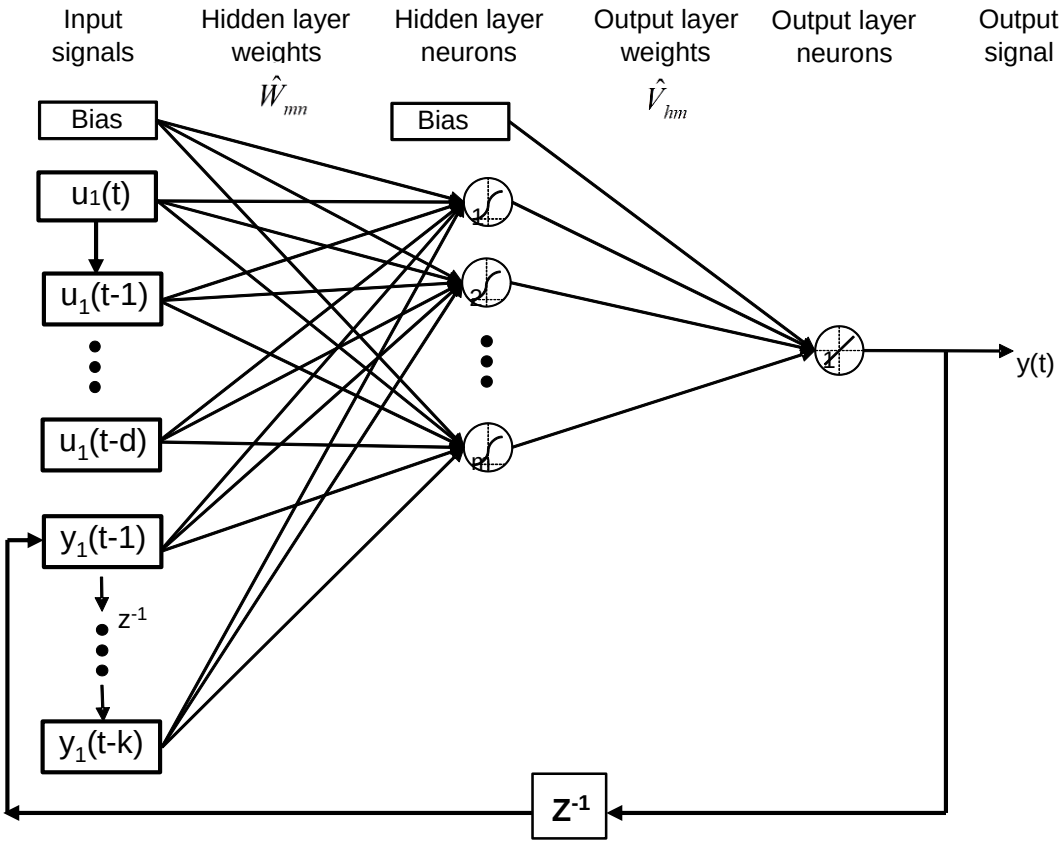

Figure 1. Multi-input, single-output (NARX) Recurrent Neural Network configured for RAGD training

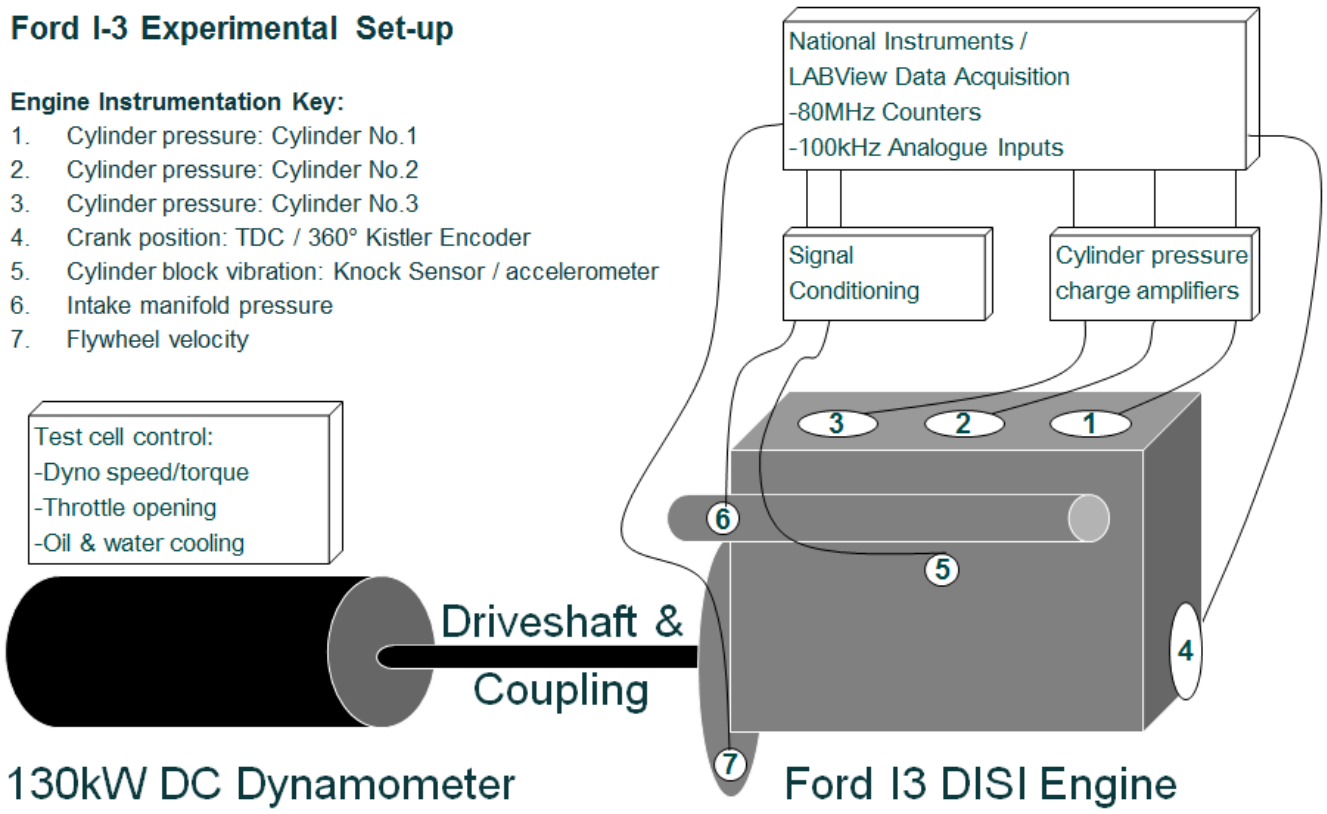

Figure 2. Layout of the measurement system connected to the Ford 3-Cylinder in-line DISI engine

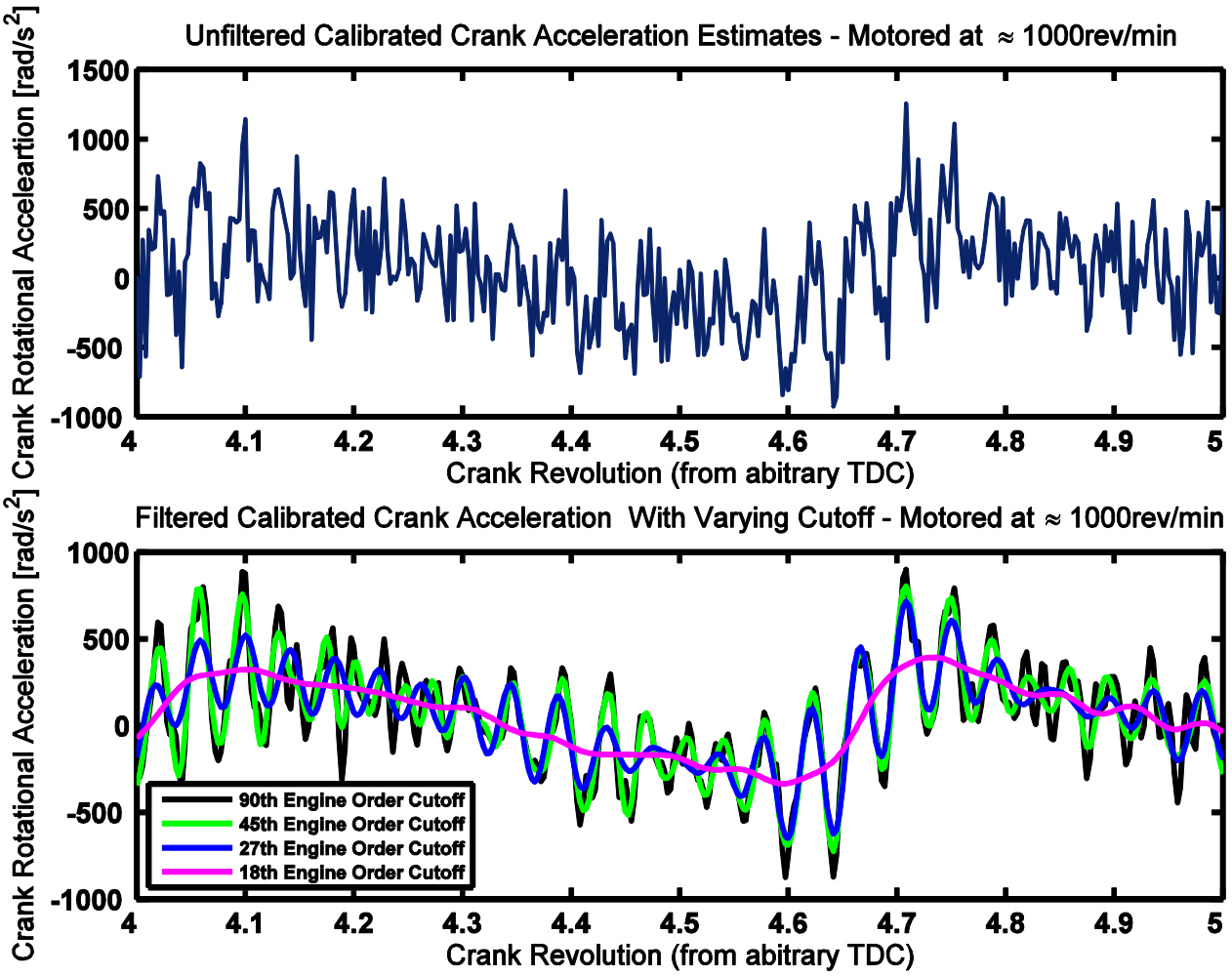

Figure 3. Measured crank acceleration derived from calibrated encoder data showing effect of noise reduction via change in filter cut-off frequency. Top: Crank acceleration derived from raw calibrated encoder data. Bottom: Crank acceleration filtered at decreasing engine orders

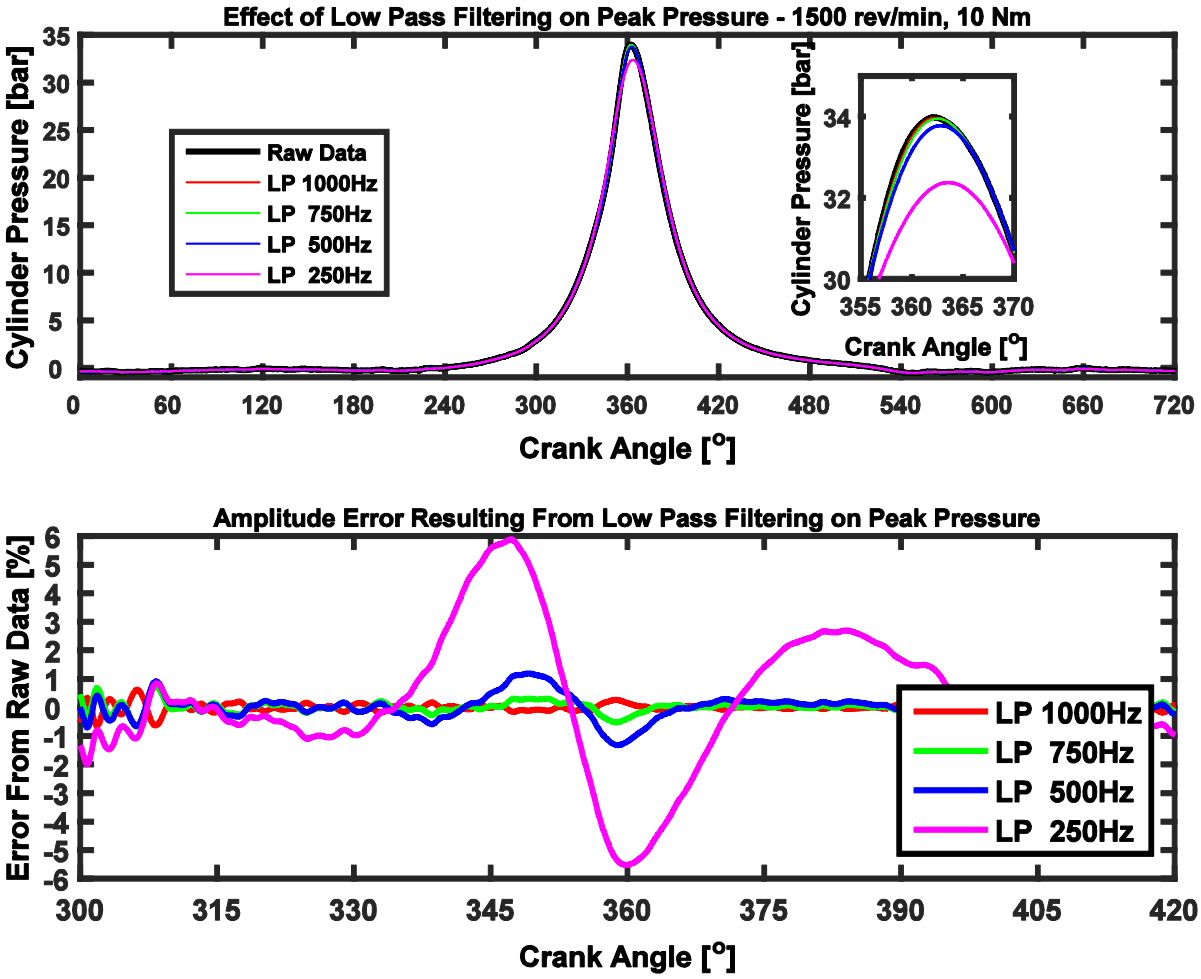

Figure 4. Effect of low pass filtering on cylinder pressure trace at 1500 rev/min, 10 Nm

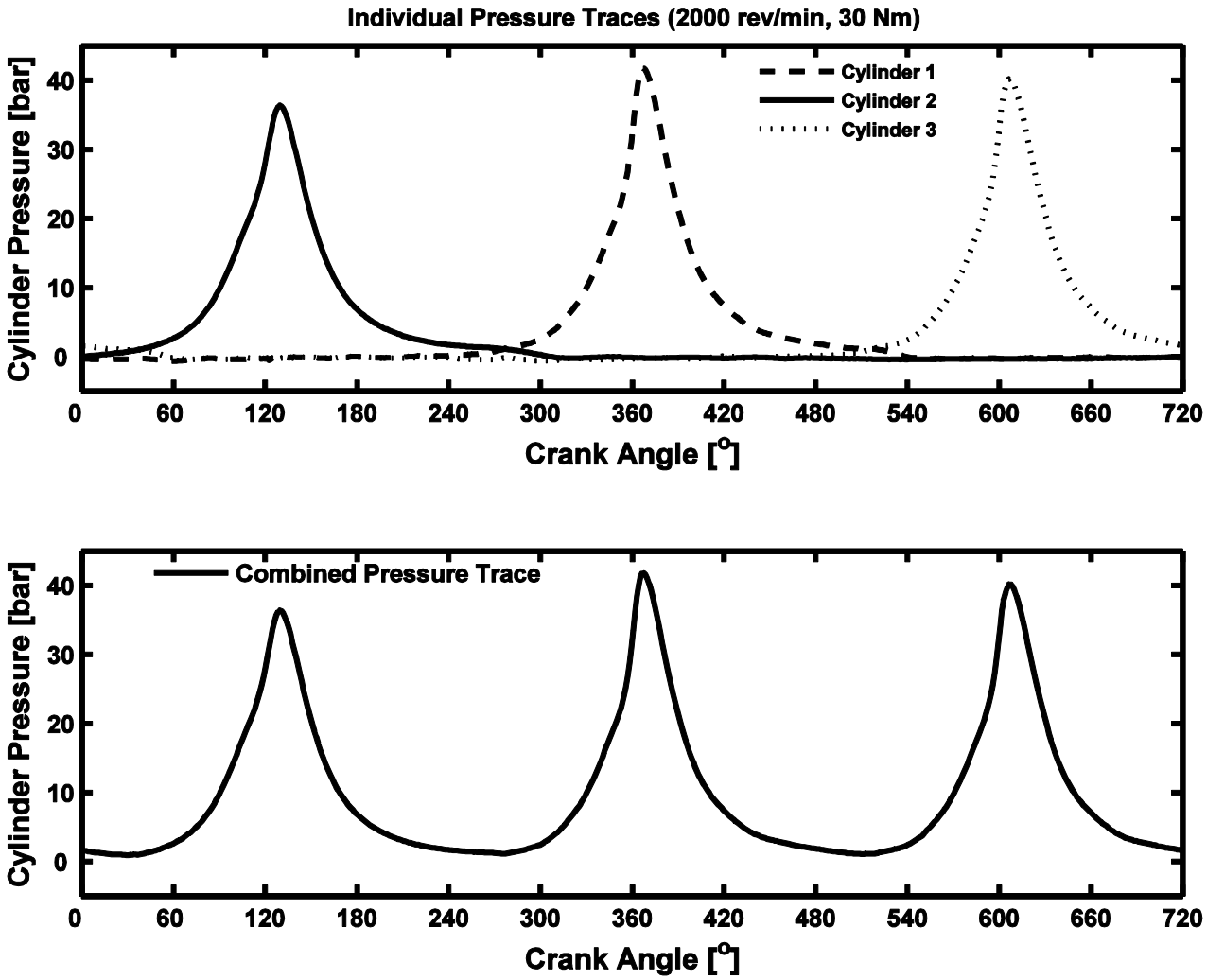

Figure 5. Measured Cylinder Pressure for three individual cylinders (Top) cylinder pressures combined into a single trace (Bottom)

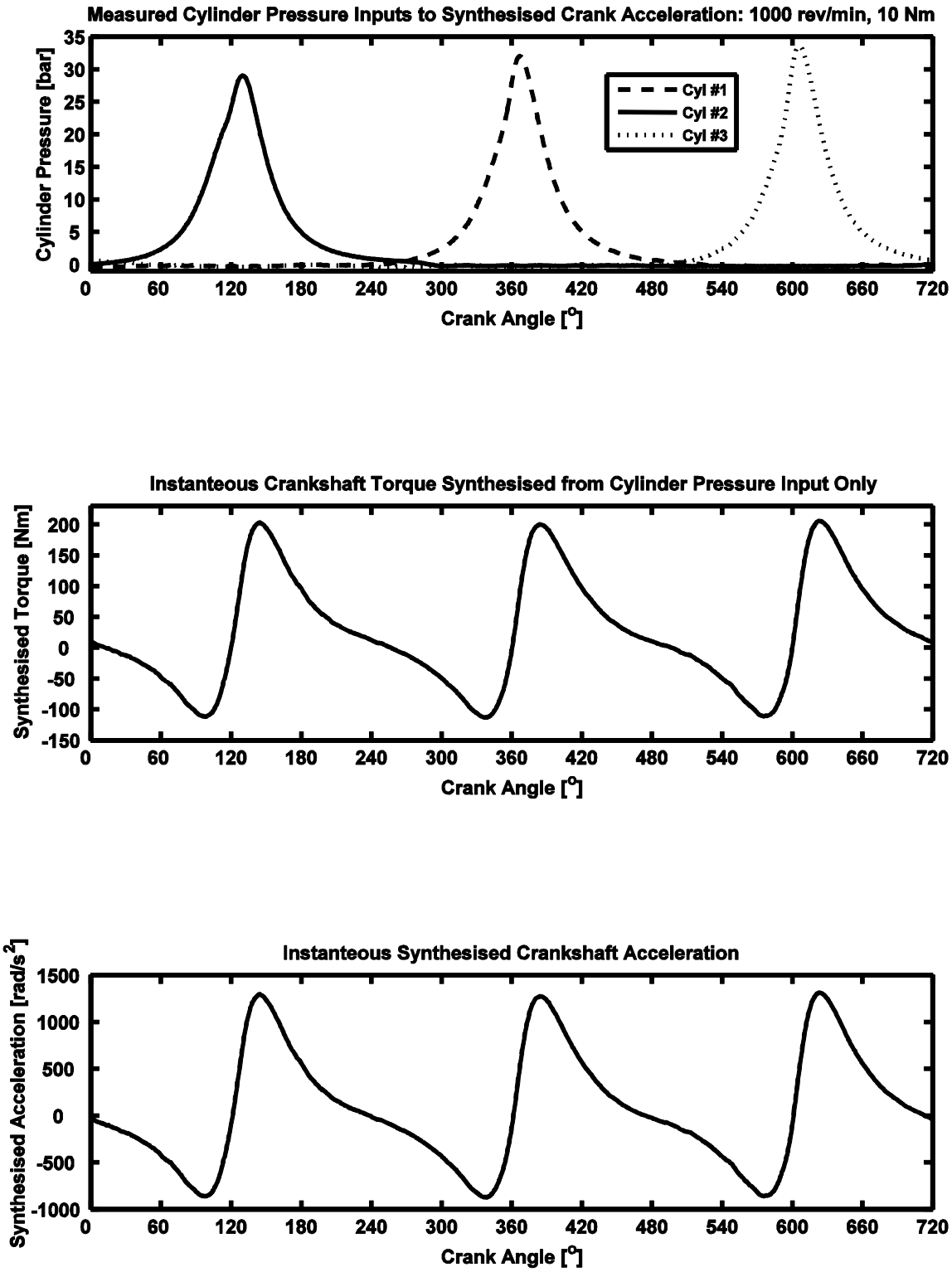

Figure 6. Synthesised crankshaft acceleration:
Top: Measured cylinder pressure;
Middle: Cylinder pressure derived crankshaft torque
Bottom: Synthesised crankshaft acceleration

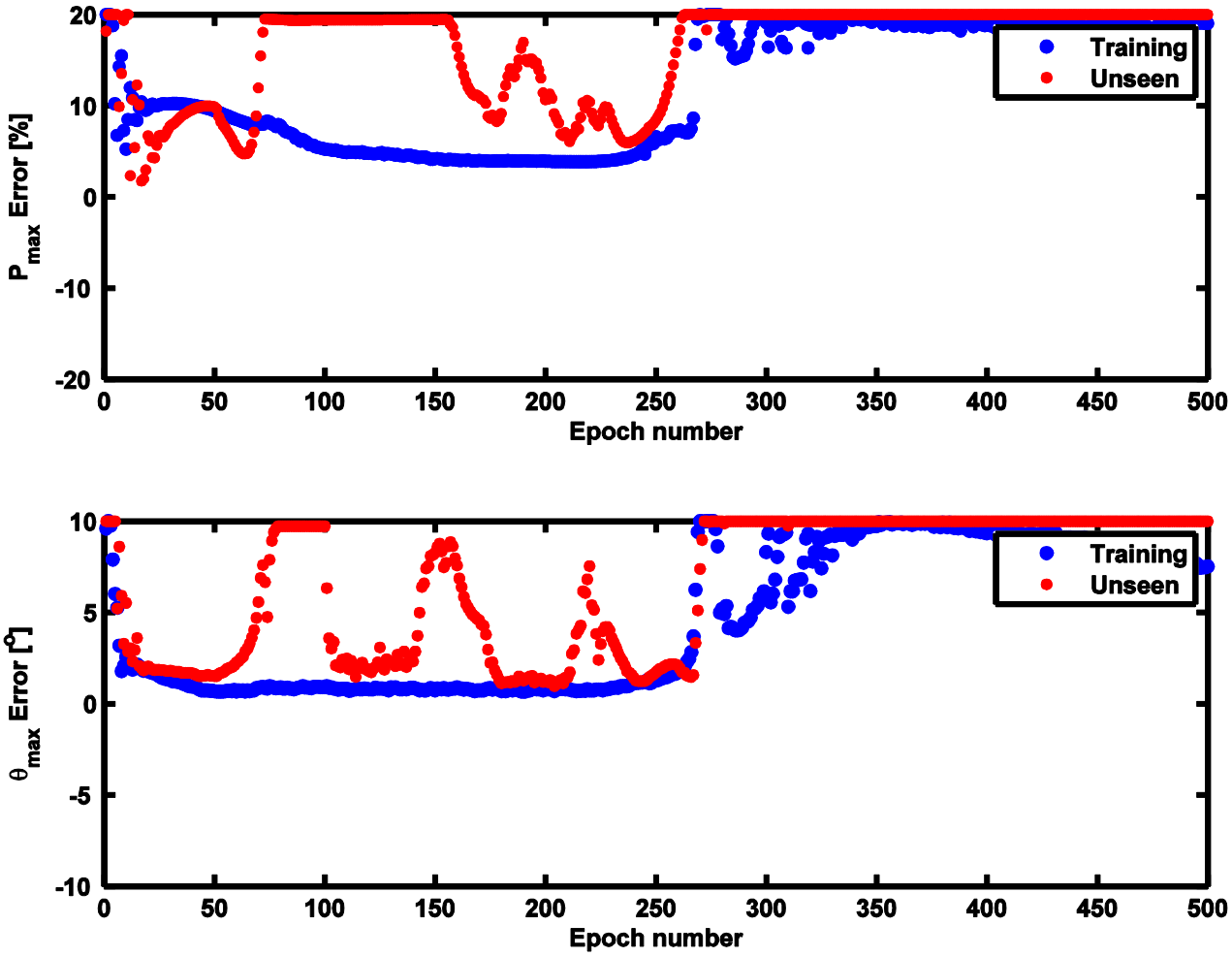

Figure 7. Errors in predicted peak pressure and location of peak pressure as a function of the number of epochs, using synthesised crank acceleration as input data for a network with 8 hidden-layer neurons for nominal engine test point: 1000 rev/min, 10 Nm.

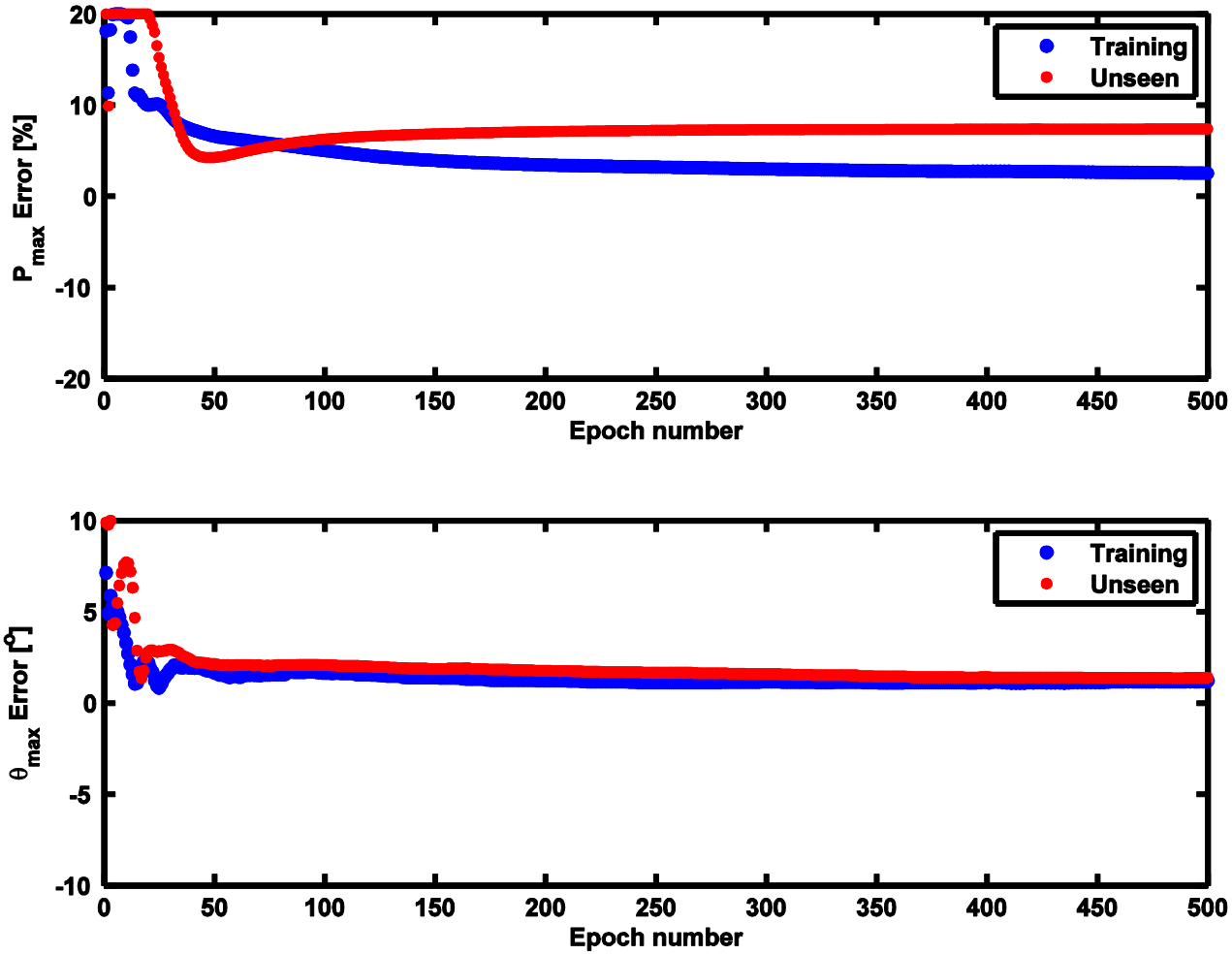

Figure 8. Errors in predicted peak pressure and location of peak pressure as a function of the number of epochs, using synthesised crank acceleration as input data for a network with 24 hidden-layer neurons for nominal engine test point: 1000 rev/min, 10 Nm.

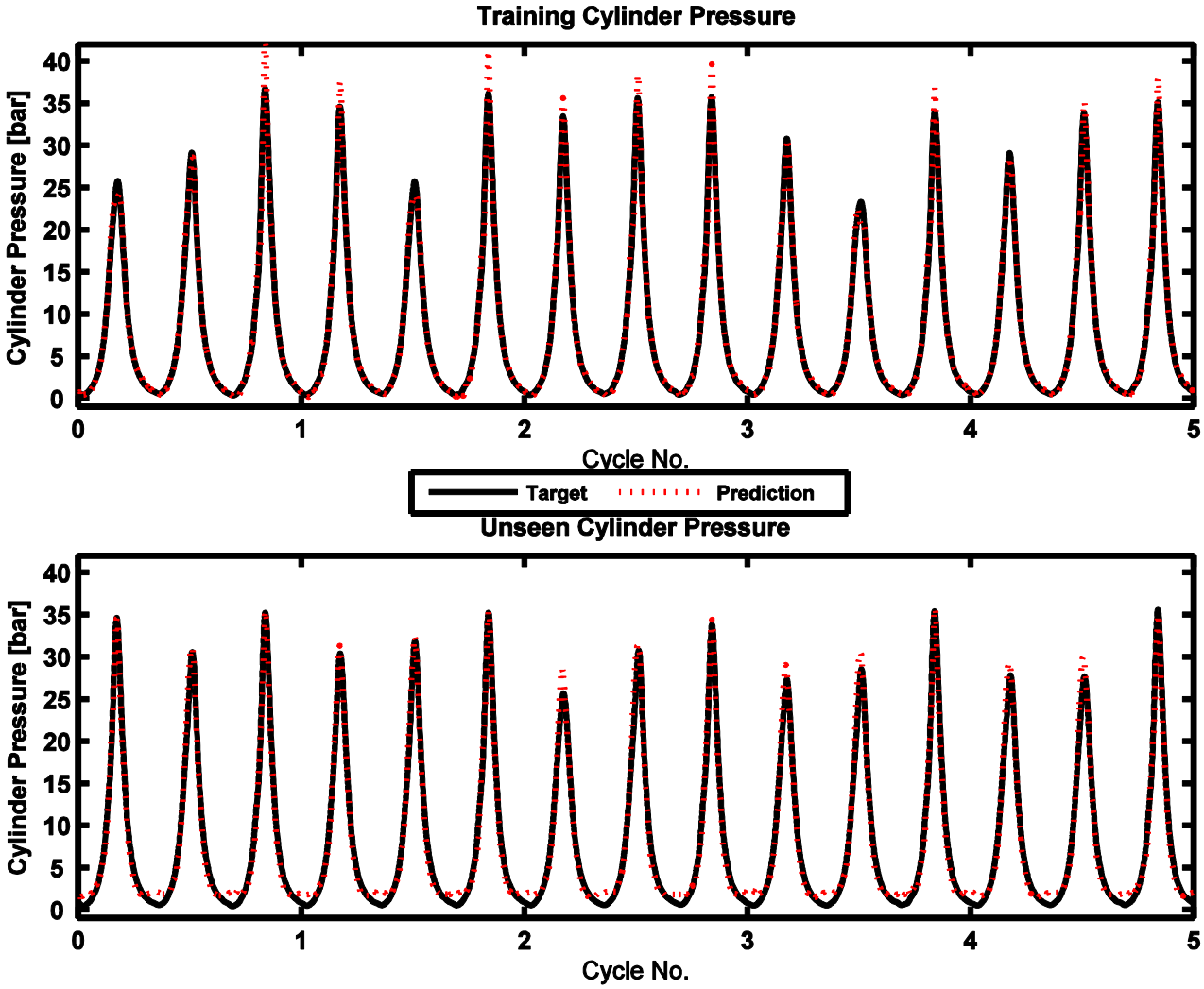

Figure 9. Predicted cylinder pressure using synthesised crank acceleration corresponding to the training data (top), and the unseen (generalisation) data (bottom) using the static weights of the 47th training epoch of a NARX network with 24 hidden-layer neurons for nominal engine test point: 1000 rev/min, 10 Nm.

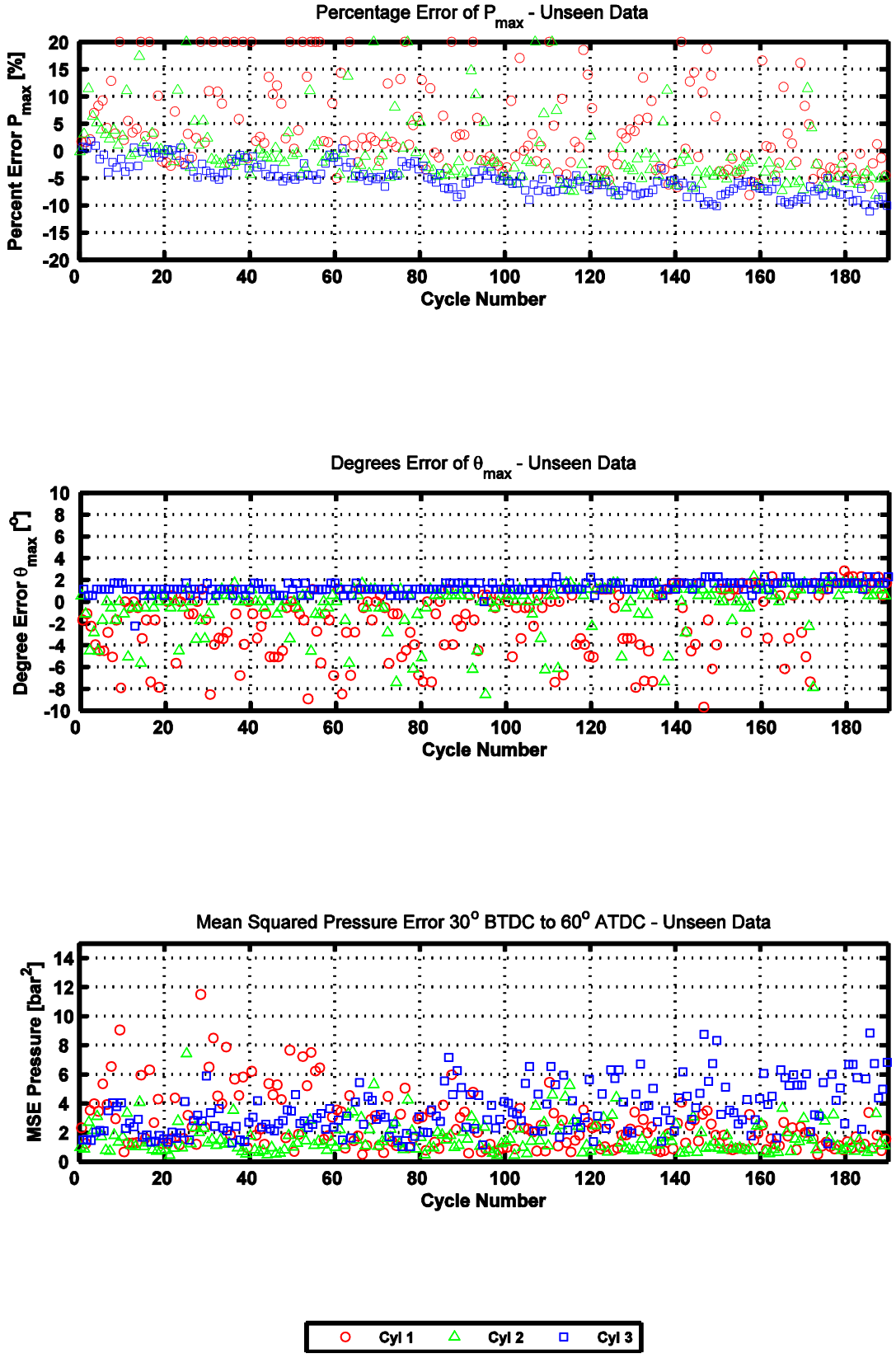

Figure 10. Errors in predicted peak pressure and location of peak pressure for RAGD training via synthesised crank acceleration using the static weights of the 47th training epoch of a NARX network with 24 hidden-layer neurons for 3 cylinders at nominal engine test point: 1000 rev/min, 10 Nm. (Bottom): Pressure MSE over window 30° before top dead centre (BTDC) to 60° after top dead centre (ATDC)

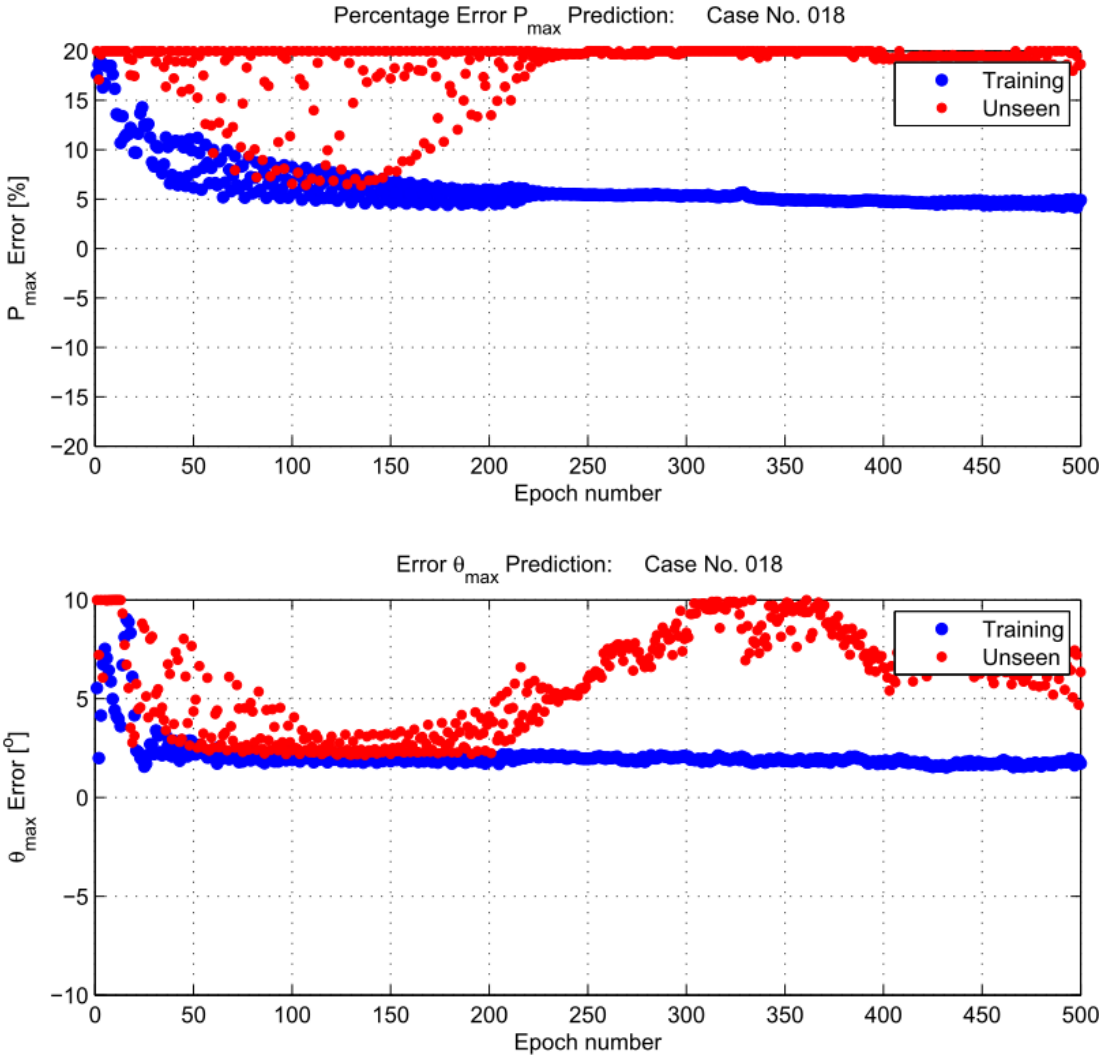

Figure 11. Errors in predicted peak pressure and location of peak pressure as a function of the number of epochs, using measured crank acceleration as input data for a network with 24 hidden-layer neurons for nominal engine test point: 1000 rev/min, 10 Nm.

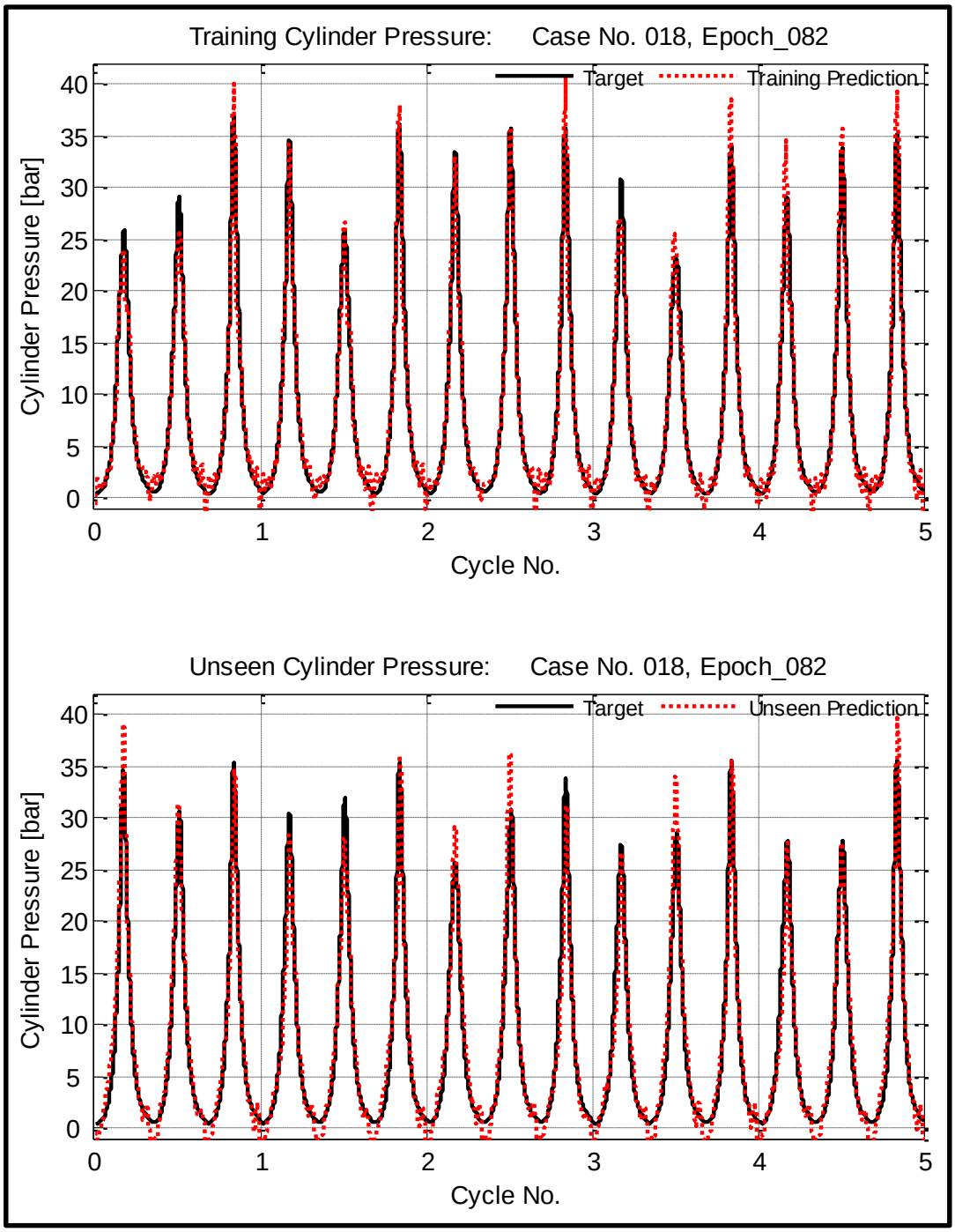

Figure 12. Predicted cylinder pressure using measured crank acceleration corresponding to the training data (top), and the unseen (generalisation) data (bottom) using the static weights of the 82nd training epoch of a NARX network with 24 hidden-layer neurons for nominal engine test point: 1000 rev/min, 10 Nm.

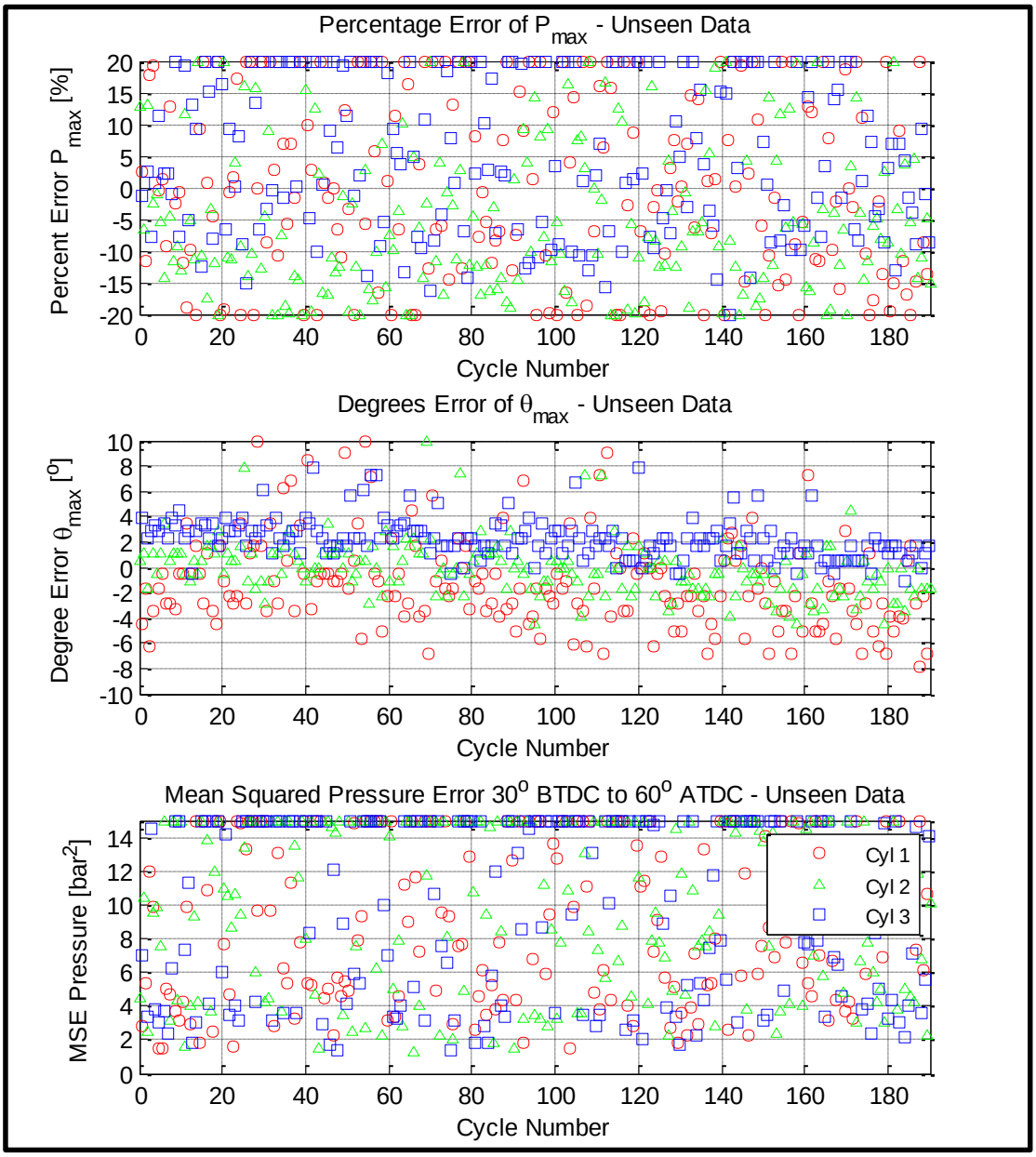

Figure 13. Errors in predicted peak pressure and location of peak pressure for RAGD training via measured crank acceleration using the static weights of the 82nd training epoch of a NARX network with 24 hidden-layer neurons for 3 cylinders at nominal engine test point: 1000 rev/min, 10 Nm. (Bottom): Pressure MSE over window 30° BTDC to 60° ATDC

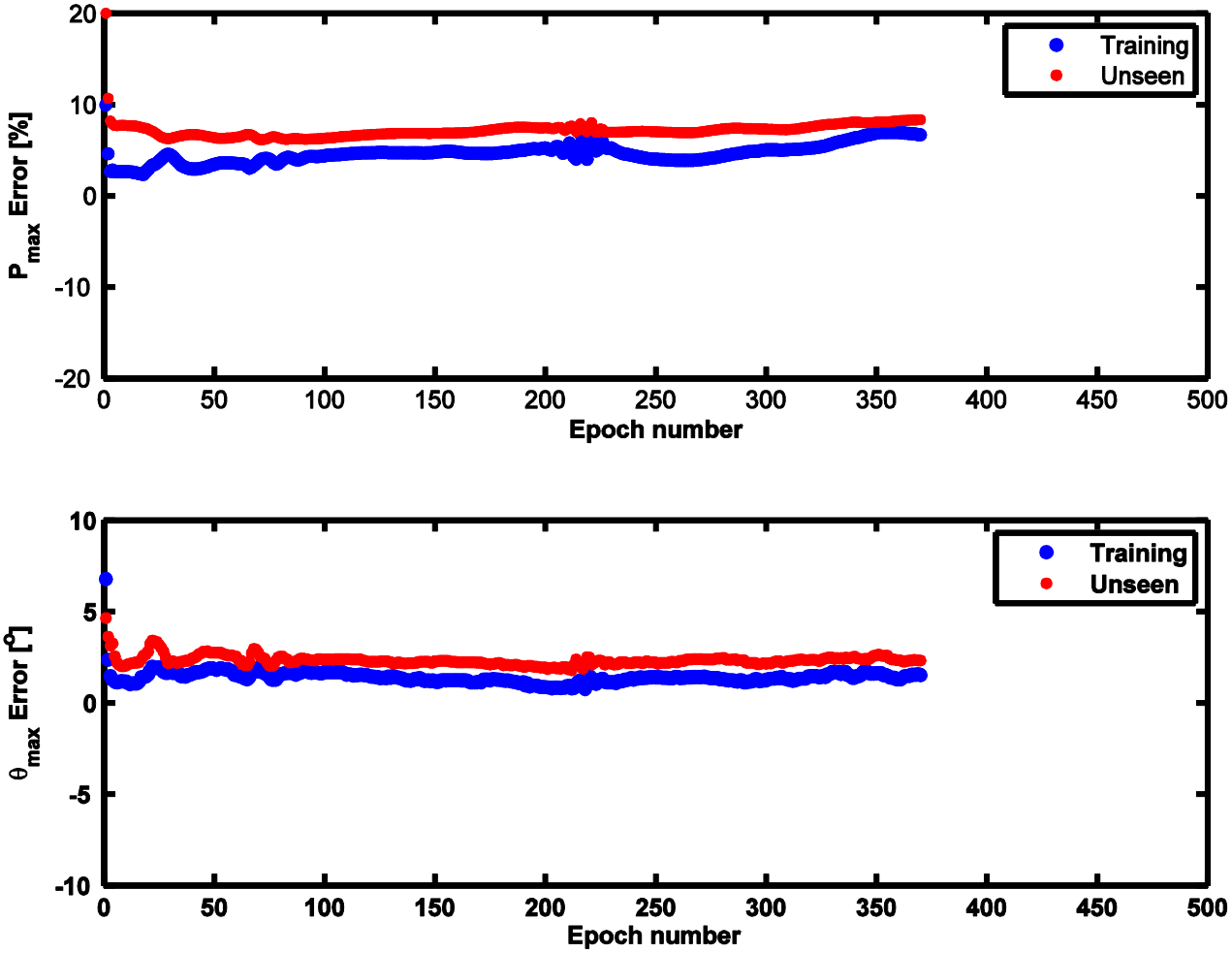

Figure 14. Errors in predicted peak pressure and location of peak pressure as a function of the number of epochs, using 10 cycles of both measured crank velocity and acceleration as input data for a network with 24 hidden-layer neurons of nominal engine test point: 1000 rev/min, 10 Nm.

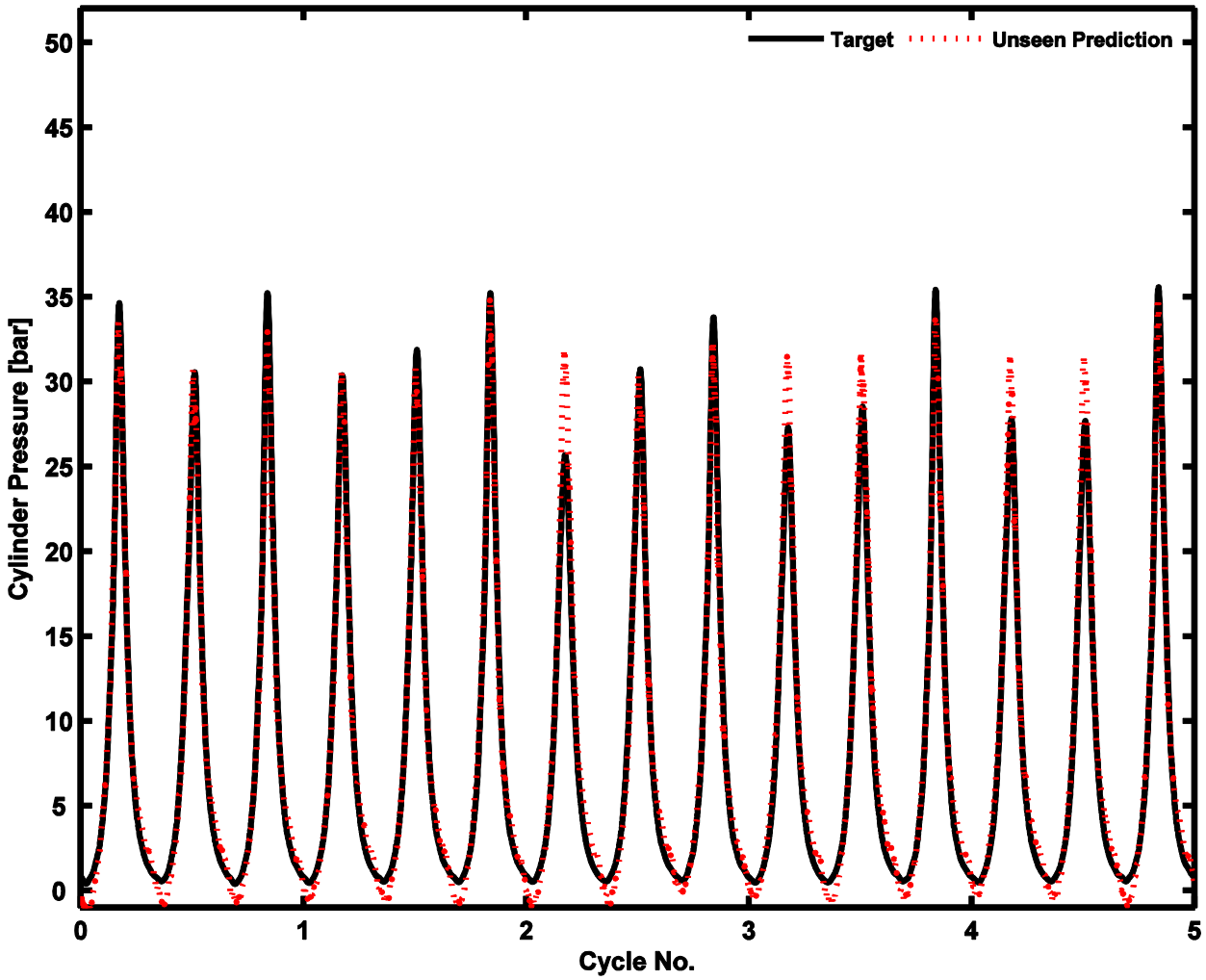

Figure 15. Predicted cylinder pressure using measured crank velocity and acceleration corresponding to the unseen (generalisation) data using the static weights of the 27th training epoch of a NARX network with 24 hidden-layer neurons for nominal engine test point: 1000 rev/min, 10 Nm.

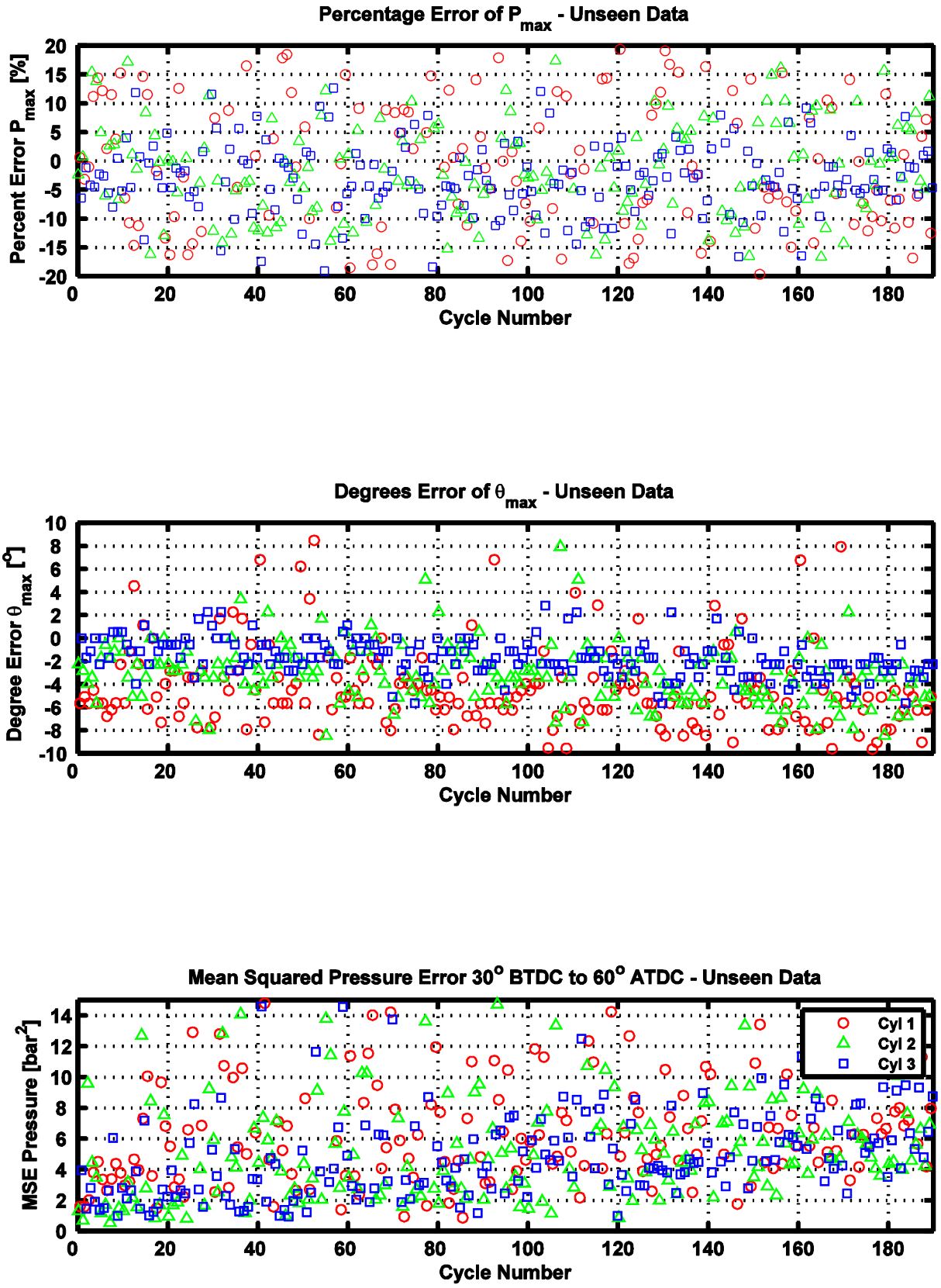

Figure 16. Errors in predicted peak pressure and location of peak pressure for RAGD training via measured crank velocity and acceleration using the static weights of the 27th training epoch of a NARX network with 24 hidden-layer neurons for 3 cylinders at nominal engine test point: 1000 rev/min, 10 Nm. (Bottom): Pressure MSE over window 30° BTDC to 60° ATDC